ResearchSouth, Inc.

software development and applications in
mechanical engineering

555 Sparkman Dr.
Suite 1612
Huntsville, AL 35816
Phone 256 721-1769
Mobile 256 603-5596
Fax 256 430-3278
lawrence@researchsouthinc.com
www.researchsouthinc.com

Contract NNL08AA33C

Multidisciplinary Modeling Software for Analysis, Design, and Optimization of HRRLS Vehicles

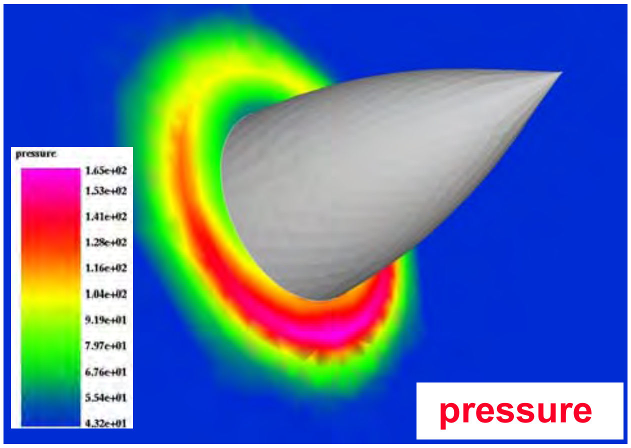
Final Report
(Deliverable #40)

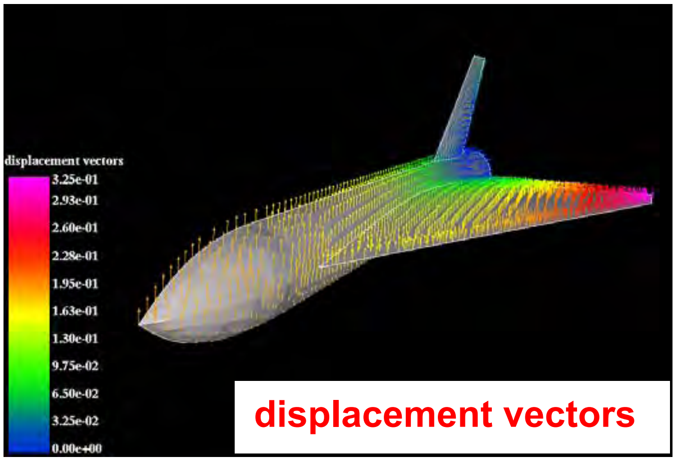
March 2011

Lawrence W. Spradley, PhD   Principal Investigator
Rainald Lohner, PhD
James L. Hunt

Submitted To: **Kim S. Bey, PhD**
NASA Langley Research Center
Hampton, VA 23681-2199

Coupled RANS and FEA





FOREWORD

This Final Report was prepared for NASA-Langley contract NNLO8AA33C, entitled "Multidisciplinary Modeling Software for Analysis, Design, and Optimization of HRRLS Vehicle." The work described herein was conducted by ResearchSouth, Inc., Huntsville, Alabama.

The Principal Investigator for this contract is Dr. Lawrence W. Spradley. Consulting support is provided by Dr. Rainald Lohner and James L. Hunt.

The NASA-Langley Technical Monitor is Dr. Kim S. Bey.

All inquiries regarding this Report should be directed to:

Lawrence W. Spradley, Ph.D.
ResearchSouth, Inc.
555 Sparkman Drive
Suite 1612
Huntsville, AL 35816
Telephone: (256) 721-1769
FAX: (256) 430-3278
e-mail: lawrence@researchsouthinc.com

CONTENTS

1.0 TECHNICAL APPROACH

1.1 Background

The concept for Highly Reliable Reusable Launch Systems (HRRLS) under the NASA Hypersonics project is a two-stage-to-orbit, horizontal-takeoff / horizontal-landing, (HTHL) architecture with an air-breathing first stage. The first stage vehicle is a slender body with an air-breathing propulsion system that is highly integrated with the airframe. The light weight slender body will deflect significantly during flight. This global deflection affects the flow over the vehicle and into the engine and thus the loads and moments on the vehicle. High-fidelity multidisciplinary analyses that accounts for these fluid-structures-thermal interactions are required to accurately predict the vehicle loads and resultant response. These predictions of vehicle response to multi-physics loads, calculated with fluid-structural-thermal interaction, are required in order to optimize the vehicle design over its full operating range.

This contract with ResearchSouth addresses one of the primary objectives of the Vehicle Technology Integration (VTI) discipline: the development of high-fidelity multi-disciplinary analysis and optimization methods and tools for HRRLS vehicles. The primary goal of this effort is the development of an integrated software system that can be used for full-vehicle optimization. Over a three year period, ResearchSouth will:

1) integrate the master code, FEMAP, into the multidiscipline software network to direct the coupling to assure accurate fluid-structure-thermal interaction solutions:

2) loosely-couple the Euler flow solver FEFLO to the available and proven aeroelasticity and large deformation (FEAP) code,

3) provide a coupled Euler-boundary layer capability for rapid viscous flow simulation,

4) develop and implement improved Euler/RANS algorithms into the FEFLO CFD code to provide accurate shock capturing, skin friction, and heat-transfer predictions for HRRLS vehicles in hypersonic flow,

5) perform a Reynolds – averaged Navier – Stokes computation on an HRRLS configuration,

6) integrate the RANS solver with the FEAP code for coupled fluid –structure – thermal capability,

7) integrate the existing NASA SRGULL propulsion flow path prediction software with the FEFLO software for quasi-3D propulsion flow path predictions,

8) improve and integrate into the network, the existing adjoint-based design optimization code

9) investigate uncertainty methods for fluid – structure interactions,

10) couple the multi-discipline code to six degree of freedom rigid body motion

This Final Report presents our technical approach in Section 1. It is then organized to discuss the above nine topics in self-contained sections with references and recommendations included there.

1.2 Computer Software

The ResearchSouth plan is to provide a multi-disciplinary software package to NASA. The tool will be based on first principles and include coupled multi-physics, coupled disciplines, and code interfaces. There are six main existing components that we used to build the software tool plus our existing master, executive code FEMAP. The following are the code components for our master plan for producing the new multi-disciplinary modeling tool.

Executive	**FEMAP**	**Master Control Code**
Component #1	**FEFLO**	**Euler and RANS CFD module**
Component #2	**FEAP**	**Stress analysis, displacements** **Structural dynamics, large deformations**
Component #3	**HUD**	**Integral boundary layer solver**
Component #4	**FEHEAT**	**Heat transfer solver**
Component #5	**SRGULL**	**Combustion cycle analysis**
Component #6	**FEOPT**	**CFD-based design optimization** **Uncertainty Analysis**

The codes are now briefly described.

FEFLO – A Computational Fluid Dynamics Tool [1]

Authors: Rainald Lohner, Lawrence Spradley, Others
George Mason University, ResearchSouth, Inc.

The CFD software package that was used in this project has been developed for modeling of low speed incompressible and high-speed compressible fluid flow. It has been developed to operate virtually independent of computer platform or operating system. This software consists of the following modules: 1) a stand-alone CAD geometry modeling package, 2) an automatic mesh generation module for unstructured tetrahedra, 3) an adaptive mesh Euler / Navier-Stokes code for aerodynamic flow analysis, and 4) a post processing graphics code. Each of these modules is now briefly described.

The stand-alone CAD-based solid geometry modeling software generates files in the format required by the FEFLO mesh module. The user can create geometries by input of points, lines and surfaces. Import of existing CAD data from commercial packages can also be done. Mesh development is done with the GEN3D module. This unstructured tetrahedral mesh generator uses the advancing front algorithm and provides the tetrahedral meshes required by the FEFLO flow code. This module can improve meshes by redistributing points, refining the mesh, or coarsening the mesh for smooth variations in mesh density.

Flowfields are computed using the three-dimensional unstructured mesh CFD solver FEFLO. The solutions may be graphically displayed at intermediate times or at completion of the flowfield solution using the POST3D module. This post processor includes the stand-alone FEFLO pre/post processor as well as software for accessing the commercial code TECPLOT and the open source code ParaView.

The FEFLO module is a three-dimensional unstructured mesh flow solver with adaptive mesh refinement. The flow equations include the compressible Euler and Navier-Stokes equations. The methodology includes the capability for treating moving boundaries with prescribed motion or moving bodies with motion computed from six degree of freedom classical mechanics based on aerodynamic forces. The equations of state supported by FEFLO include ideal gas, equilibrium air curve fits, and a finite rate chemistry option. In order to compute flows with moving bodies, the equations are solved in the Arbitrary Lagrangean-Eulerian (ALE) coordinate frame.

The spatial discretization is accomplished on unstructured tetrahedral grids. In order to achieve high execution speeds, edge-based data structures are used. Both central and upwind flux (van Leer, Roe) formulations are included. For the temporal discretization, both Taylor-Galerkin and Runge-Kutta explicit time integration schemes are available. A fully implicit scheme is also available which uses a pre-conditioned Lower-Upper Symmetric Gauss-Seidel (LU-SGS) or a GMRES algorithm. Monotonicity of the solution is achieved through a blend of second and fourth order dissipation for the explicit schemes, Flux-Corrected Transport (FCT), or classic TVD limiters for the flux split methods. The most successful and effective iterative method which we use is LU-SGS for inviscid flows and GMRES with an incomplete lower-upper factorization preconditioner for RANS.

The generation of isotropic (Euler) unstructured grids has reached a fairly mature state, as evidenced by the many publications that have appeared over the last decade on this subject and the widespread use of unstructured grids in industry. The two most widely used techniques are the advancing front technique and the Delaunay triangulation. Hybrid schemes that combine an advancing front point placement with the Delaunay reconnection have also been used successfully. These isotropic mesh generation techniques tend to fail when attempting to generate highly stretched elements, a key requirement for Reynolds-Averaged Navier Stokes (RANS) calculations with turbulence models that reach into the sublayer.

Any reliable RANS-based solver requires a proper anisotropic or BL mesh to accurately approximate the speed profile close to the body. When generating a BL mesh, the first difficulty is to deal with the very high aspect ratios of the elements, which are of the order 10^3 to 10^5 as the width of the boundary layer depends on the local Reynolds number. Thus far, the generation of BL meshes has been carried out by an extrusion of the initial surface along the normals to the surface or by local modification of the mesh. Using only information from normals requires several improvements to obtain a quality mesh and a smooth transition to the inviscid, isotropic mesh region, particularly for complex surfaces or close to cusps or ridges. These techniques rely on a complex pre-processing of the surface in order to extract additional geometric information as convex or concave ridges. An important difficulty arises once the BL process is finished. Indeed, in most methods, a global mesh generation process is used to close the volume.

Both the Delaunay and the Advancing Front methods are susceptible to fail in the presence of anisotropic faces. In these methods, the BL mesh generation is performed in a pre-processing procedure that is done prior to any computation. Consequently, achieving a full coupling between RANS mesh generation and mesh adaptation is tedious and remains a challenge. In many applications, adapting the surface is mandatory for an accurate result. As a result, the way classical BL mesh generation works requires major changes in order to be used together with anisotropic mesh adaptation. The goal of our work is to deal with the adaptation and generation of the BL mesh in a unified and generic fashion. This new issue allows us to free the problem from some of its initial difficulties: failure for closing the volume mesh and complex pre-preprocessing (e.g. as multi-normals) for complex surfaces, etc.

In our work, anisotropic mesh generation is viewed within the metric framework which thus far has been used exclusively in anisotropic mesh adaptation. More precisely, the BL mesh generation is based on a continuous field, the distance to the body, from which a proper metric field is deduced. An adaptive strategy is then devised to recover features by local modifications the boundary layers. This innovative view for generating BL meshes makes the entire procedure more generic and much more robust. One of the main assets is that the method is then able to generate, for complex geometries, a boundary layer while keeping intact the previous adaptation procedures from non-viscous simulations. This leads to a natural coupling between BL mesh generation and anisotropic mesh adaptation.

The FEFLO code has the capability to simulate three dimensional compressible turbulent flows using a choice of models. The algebraic Baldwin-Lomax model or the turbulent kinetic energy (TKE) model on unstructured meshes are commonly used. The numerical algorithm for the tke model is based on a fully coupled approach, where both the flow equations and turbulence equations are solved simultaneously. The governing equations for kinetic energy and dissipation are derived under the assumption of a large Reynolds number flow. Modifications are then made for compressibility corrections, specialized limiters, nonlinear /anisotropic extensions, near wall variants, low Re, and variable Prandtl and Schmidt numbers.

These equations are not valid in the laminar sublayer close to a solid wall where molecular effects become dominant. The most common practice to avoid integrating the turbulence equations in these regions is to make use of wall functions. The current FEFLO code has different wall functions for modeling low speed, transonic, and supersonic flows. While such an approach is useful for a large class of wall-bounded flows, it is nevertheless limited to flows where a law of the wall region exists, and thus is not valid for separated flows. In order to be able to compute both attached and separated turbulent flow problems, a two-layer approach has been implemented in the FEFLO code. The idea is to couple the tke model with a one-equation model in an automatic way to incorporate the low turbulent Reynolds number effects through the viscous sublayer. Such a two-layer approach can be used to compute both the attached and separated flow up to the wall with no empiricism.

HUD – Boundary Layer Solver [2]

Authors: NASA Langley Research Center

The HUD code [2] is a well known solver at Langley which was developed to compute viscous boundary layer effects over 2D (axisymmetric or planar) surfaces utilizing prescribed boundary layer edge conditions as inputs. Typically, the edge conditions are determined through simpler methodologies. The HUD solution proceeds by marching downstream in the direction of flow. This code has enjoyed considerable success for over 20 years due to its accuracy and reliability for the 2D problems which it was developed to address.

The procedure developed in this contract combines the 3D inviscid CFD capability of FEFLO to compute flowfields outside of the boundary layer with HUD to compute boundary layer viscous effects over vehicle surfaces. The inviscid FEFLO output is used to provide boundary layer edge conditions to HUD. This provides a reasonable merging of the accuracy of FEFLO for 3D flowfield computations and the accuracy and reliability of HUD for viscous computations. This requires that FEFLO be run first to provide inviscid flowfield data over the vehicle surface. The FEFLO data is then processed to compute flow lines over the surface with flow properties along these flow lines. These data lines are then input into HUD to compute viscous effects (heat transfer and shear) over the vehicle surface. The resulting viscous computations are based on an assumed 2D (axisymmetric or planar) character for flow along the flow lines.

In this way, the aerodynamic coefficients lift and drag can be computed accurately and, in addition, skin friction and heat transfer can be determined. This approach provides a viscous flow analysis capability over full vehicle configurations with an efficient and practical design tool.

SRGULL - Propulsion Flow Path Code [3, 4]

Authors: NASA Langley Research Center

An efficient engineering design tool has been developed by NASA engineers for the prediction of airframe-integrated ramjet cycle performance. The SRGULL code has successfully merged elements from CFD, integral boundary layer analysis, and combustion theory to provide essentially two-dimensional nose-to-tail performance analysis of hypersonic vehicles. Use of this code has proved instrumental in the preliminary design phase of such vehicles. A space marching Euler solver integrated within the code provides the necessary flow field data which yields estimates of vehicle loads, mass capture and nozzle performance. An integro-differential boundary layer technique permits estimation of vehicle heat transfer as well as additional vehicle force and moments due to skin friction. This module is termed HUD [2]. The combustor methodology employed within the code uses a multi-step, one-dimensional computation and is applicable to both supersonic and dual-mode combustion. A program was sponsored by NASA-Langley to integrate the three-dimensional CFD code FEFLO and the baseline SRGULL code [3, 4].

Software interface routines have been developed to automatically pass the CFD external flow field data to the SRGULL code to provide a three dimensional upstream flow solution at the inlet face. This internal flow is picked-up by the CFD code at the combustor exit, after internal flow and combustion cycle analysis is performed by SRGULL, to proceed through internal nozzle and out into external nozzle where shear layer with external flow is also calculated.

FEAP - A Finite Element Analysis Program [5]

Author: Robert Taylor
Department of Civil Environmental Engineering
University of California at Berkeley

FEAP is a Multi-Dimensional finite element code for structural, continuum, solid mechanics, heat transfer, and coupled field simulations. It is very flexible with static and dynamic time integration and steady state capabilities and modular with a rudimentary front and backend. It is written mostly in FORTRAN except for the backend graphics and the dynamic memory management system. This is a code for people interested in performing finite element code development research and those with particularly unusual problems that can not be handled with commercial codes.

During the last several decades, the finite element method has evolved from a linear structural analysis procedure to a general technique for solving non-linear, transient, partial differential equations. An extensive literature on the method exists which describes the theory necessary to formulate solutions for general classes of problems, as well as, practical guidelines in its application to problem solution.

The manual describes many of the features of the general purpose Finite Element Analysis Program (FEAP) to solve such problems. Many of the descriptions in the manual are directed to the solution of problems in solid mechanics; however, the system may be extended to solve problems in other subject areas by adding user developed modules to address a specific class of new problems. Such extensions have been made by users to solve problems in fluid dynamics, flow through porous media and thermo-electric fields [5]. Interested readers are directed to the FEAP Programmers Manual for details on adding new features.

The finite Element Analysis Program (FEAP) is a computer analysis system designed for:

- Use in course instruction to illustrate performance of different types of elements and modeling methods;

- In a research, and/or applications environment which requires frequent modifications to address new problem areas or analysis requirements.

The computer system may be used in either a UNIX/Linux or a Windows environment and includes an integrated set of modules to perform.

- Input of data describing a finite element model
- Construction of solution algorithms to address a wide range of application; and
- Graphical and numerical output of solution results.

A problem solution is constructed using a command language concept in which the solution algorithm is completely written by the user. Accordingly, with this capability, each application may use a solution strategy which meets specific needs. There are sufficient commands included in the system for linear and non-linear applications in structural or fluid mechanics, transfer. And many other areas requiring solution of problems modeled by differential equations; including those steady state and transient problems.

Users also may add new routines for mesh generation and manipulation; model element or material description; new command language statements to meet specific application requirements; and plot outputs for added graphical display. These additions may be used to assist generation of meshes for specific classes of problems; to import meshes generated by other systems; or to interface with other graphical devices.

The current FEAP system contains a general element library. Elements are available to model one, two or three dimensional problems in linear and/or non-linear structural and solid mechanics and for linear heat conduction problems. Each of the provided solid elements provides access to a material model library. Material models are provided for elasticity, visoelasticity, plasticity, and heat transfer constitutive equations. Elements also provide capability to generate mass and geometric stiffness matrices for structural problems and to compute output quantities associated for each element (e.g., stress, strain); including capability of projecting these quantities to nodes to permit graphical outputs of the results contours.

Users also may add an element to the system by writing and linking a single modular to the FEAP system. Details on specific requirements to add an element as well as other optional feature available are included in the FEAP programmers Manual. This manual describes how to use many of the excising capabilities in the FEAP system [5].

1.3 Loosely Coupled Approach

One of the main aims of the loosely coupled approach is to integrate the different codes such that each one is modified in the least possible way. Moreover, the option of having different grids for different disciplines, as well as adaptive grids that vary in time, implies that in most cases no fixed common variables will exist at the boundaries. Therefore, fast and accurate interpolation techniques are required. As the grids may be refined or coarsened during time steps, and the surface deformations may be severe, the interpolation procedures have to combine speed with generality.

The CFD data for flow cases consists of density, velocity and energy values assigned to the CFD mesh points and loaded onto a body surface dump file. The mesh elements are triangular face elements which cover the body surface. The body geometry data, including point coordinates and element connectivities are also loaded onto the dump file. The CFD data on the file are first decoded for conversion to pressures at the mesh points. These point data are then transposed to the structural surface elements which contain the CFD mesh points.

The basis of this coupling procedure is the existing and proven Finite Element Multidisciplinary Applications Package FEMAP [1]. The FEMAP module is a "Master" code that directs the coupling of existing codes from different multi-physics disciplines: Computational Fluid Dynamics (CFD), Computational Structural Dynamics (CSD), or Computational Thermo-Dynamics (CTD), Computational Propulsion-Dynamics (CPD) to simulate problems that exhibit significant physical couplings between fluids, structures, thermal heating, propulsion flow-path, control, etc. One of the primary objectives of our approach is to couple the different codes in such a way that each one is modified as little as possible.

The individual codes become subroutines or objects that are called from FEMAP during time-stepping. The current version of FEMAP couples several modules to provide a realistic simulation of fluid-structure interaction. These modules include CFD (FEFLO), stress analysis (FEAP), large deformation structural dynamics (FEAP) and thermal analysis (FEAP). The development of any new CFD or structural dynamics codes is not required; rather we use existing and proven modules

Moreover, the option of having different grids for different disciplines, as well as adaptive grids that vary in time, implies that in most cases no fixed common variables will exist at the boundaries. Therefore, fast and accurate interpolation techniques are required. As the grids may be refined or coarsened during time steps, and the surface deformations may be severe, the interpolation procedures have to combine speed with generality. In our terminology, we will use the word "interpolation" as referring to the process of finding, from a list of faces, the one closest to a given point, and the word "evaluation" to the actual product of shape-functions with nodal values, e.g., in order to obtain the pressure at a point on a face.

The CFD data for flow cases consists of density, velocity and energy values assigned to the CFD mesh points and loaded onto a body surface dump file. The mesh elements are triangular face elements which cover the body surface. The body geometry data, including point coordinates and element connectivities are also loaded onto the dump file. The CFD data on the file are first decoded for conversion to pressures at the mesh points. These point data are then transposed to the FEAP surface elements which contain the CFD mesh points.

The CFD mesh is typically much finer than the FEAP mesh so that each CSD element may be expected to contain multiple CFD mesh points. The process of transposing the CFD data to the FEAP element should be as nearly conservative as possible within the numerical limits imposed by mesh density. We wrote a module which distributes CFD point data to the FEAP nodes that define the FEAP element containing the CFD point. This distribution is made such that total force and moments over the element are conserved.

The Loose Coupling process exhibits the following features:

- The separate update of the CFD and CSD domains, with a transfer of variables at the interface.
- A "master surface" for a variable is selected and the variable is interpolated or projected to the other domain at the beginning of the next time step.

  The CSD surface location and velocity = the "master-grid" for displacements.
  The CFD grid = the "master-grid" for the loads (pressures, fluid shear-stresses).

- Each iterative pass is composed of the following steps:

  Solve for CFD with imposed displacements and accelerations from the CSD solution;
  Solve for CSD with imposed loads and stresses from the CFD solution;

- Variables on the boundaries are transferred between the codes by the master code (FEMAP) where each code (CFD, CSD, CTD) is seen as a subroutine, or object, called by FEMAP.
- Transfer of geometrical and physical information is performed between the different codes without affecting their layout, basic functionality, and coding styles.

This loosely-coupled procedure consists of the following steps:

Step 1: Compute flow and pressure loading with CFD code FEFLO on the structure
Step 2: Perform thermal analysis using FEAP and output thermal loads
Step 3: Transfer pressure and thermal loads to FEAP mesh to compute deformed shape
Step 4: Determine if re-meshing is necessary
Step 5: (If YES, remesh deformed shape and) re-compute dynamic loads with FEFLO
Step 6: Transfer again to FEAP to compute new stresses, dynamics and deformed shape
Loop for specified time or until steady state is reached.

Figure 1 is a block diagram of the procedure that is being used to loosely couple these fluid-structure-thermal codes.

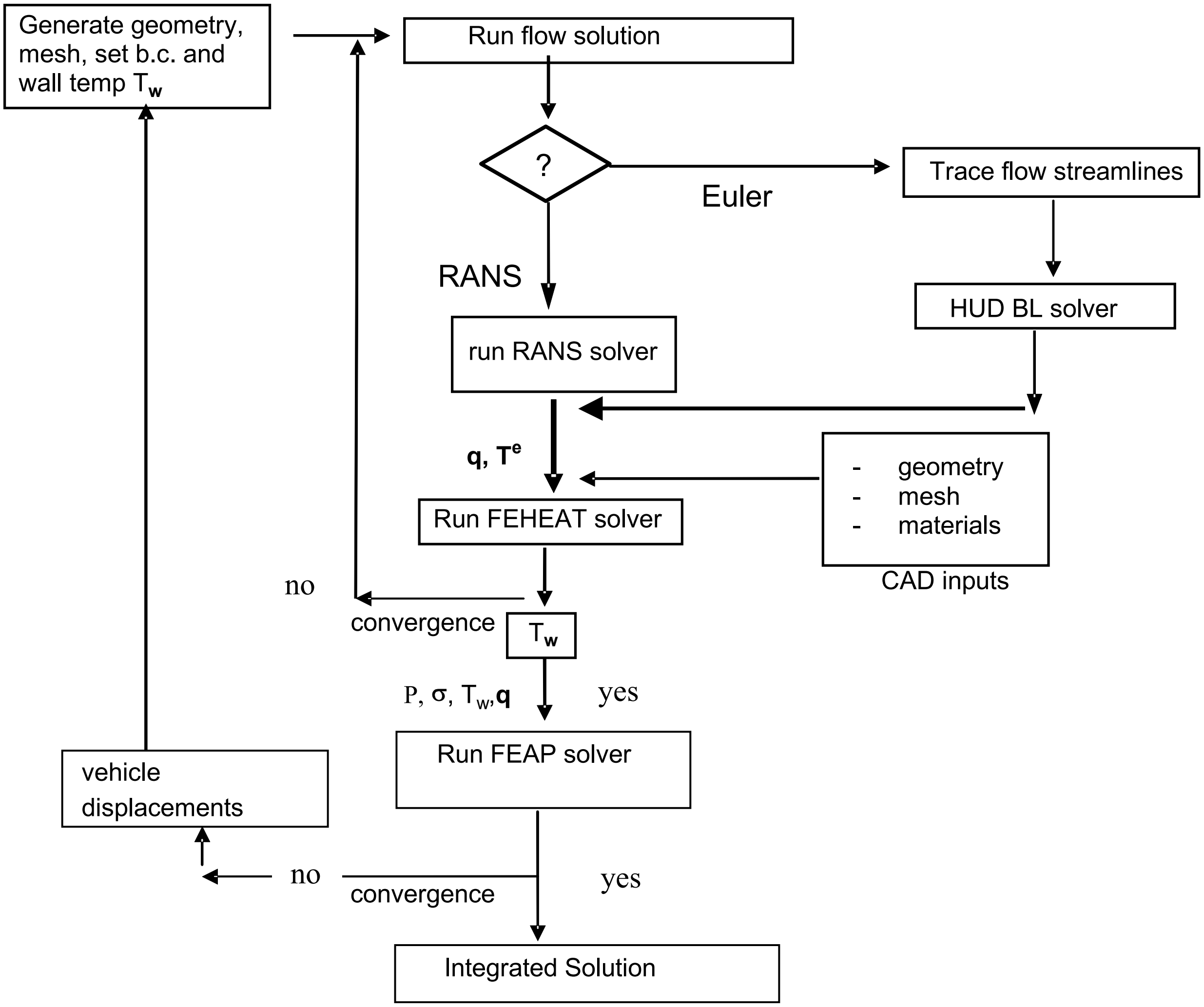


Fig. 1 Loose Coupling Procedure for Fluid – Structure – Thermal Codes

1.4 References

[1] http://www.scs.gmu.edu/~rlohner/pages/publications/

[2] Pinckney, S.Z., "Method for Predicting Compressible Turbulent Boundary Layers in Adverse Pressure Gradients", NASA TM X-2302, August 1971.

[3] Pinckney, S.Z., "Program ZMCNEW: Second Generation Engineering Model for the Prediction of Airframe-Integrated Subsonic / Supersonic Combustion Ramjet Cycle Performance".

[4] Spradley, Lawrence, Rainald Lohner, and Pinckney, S. Z., "Integration of Three-Dimensional CFD with Combustion Cycle Analysis", JANNAF Paper APS-VI-06, 35th Combustion / Airbreathing Joint Meeting, Cocoa Beach, FL, October 1999.

[5] Taylor, Robert, FEAP - Department of Civil Environmental Engineering, University of California at Berkeley, 2009 (Private Communication).

2.0 FLUID – STRUCTURE – THERMAL CODE INTEGRATION

This loosely – coupled procedure is illustrated in this section using a generic airplane configuration and structure. The run conditions are given in Fig. 2.1-1. The configuration and meshes are given in Figs. 2.1-2 and 2.1-3.

Run conditions for CFD:

density : 0.121300E+01
x-veloc.: 0.677700E+03
y-veloc.: 59.290000E+00
z-veloc.: 0.000000E+00
pressure: 1.0e+5

Euler (Ma=2, alpha=5)

Riemann solver: HLLC
Limiter : vanAlbada
Solver: LU-SGS,
5 sub iterations

nelem=393022
npoin= 76063

Run conditions for CSD:

4-noded SHELL elements
ELAStic ISOTropic 0.690000E+11 0.300000E+00
DENSity 0.100000E+01 0.270000E+04
THICk 0.0 0.020000E+00 0.000000E+00 0

nelem=3104
npoin=3215

Fig. 2.1-1 Run Conditions for Test Case

2.1 Small Deformation Steady State

The solution begins by converging the solution to steady state for the initial configuration. The pressure and Mach number contours are shown in Figs. 2.1-4 and 2.1-5 respectively. These loads are passed to the FEAP solver to compute the initial deformation and surface velocity movement. Figs. 2.1-6 and 2.1-7 are plots of the initial deformation contours and surface velocity vectors.

The loop between FEFLO and FEAP is continued for 4 iteration until the deformation change is very small. The convergence history is given in Fig. 2.1-8. The initial shape (lines) and the final (solid) position of the airplane surface is plotted in Fig. 2.1-9. For comparison of the movement, Fig. 2.1-10 shows the initial and final CFD meshes.

The final pressure contours, plotted with the surface positions, is shown in Fig. 2.1-11. The corresponding Mach number solution is shown in Fig. 2.1-12.

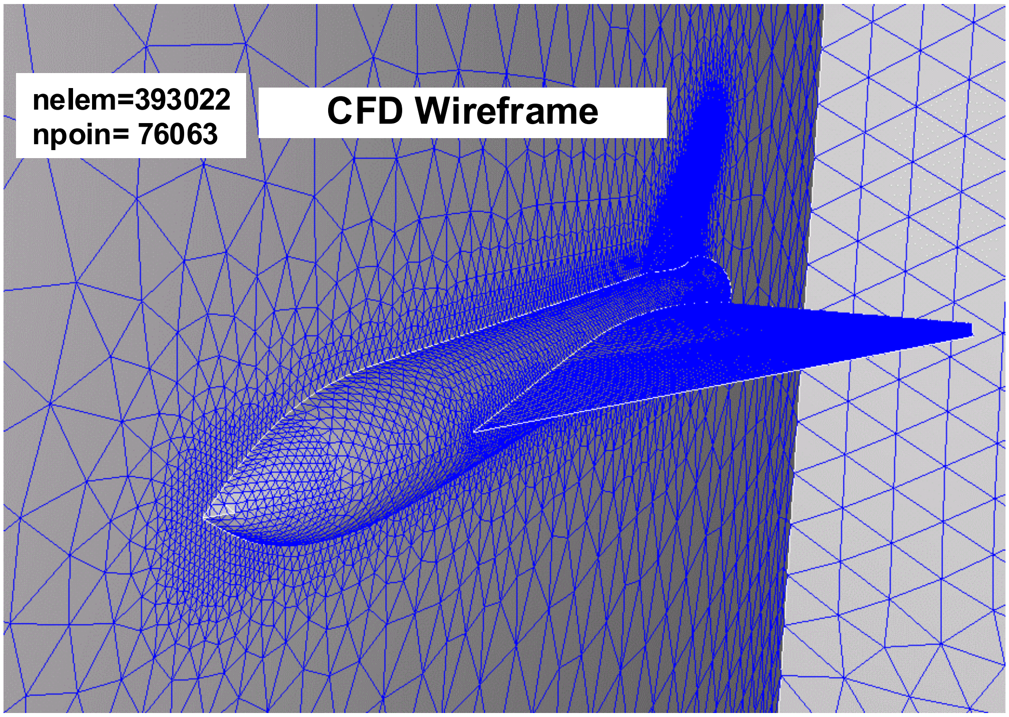


Fig. 2.1-2 CFD Mesh for Generic Airplane

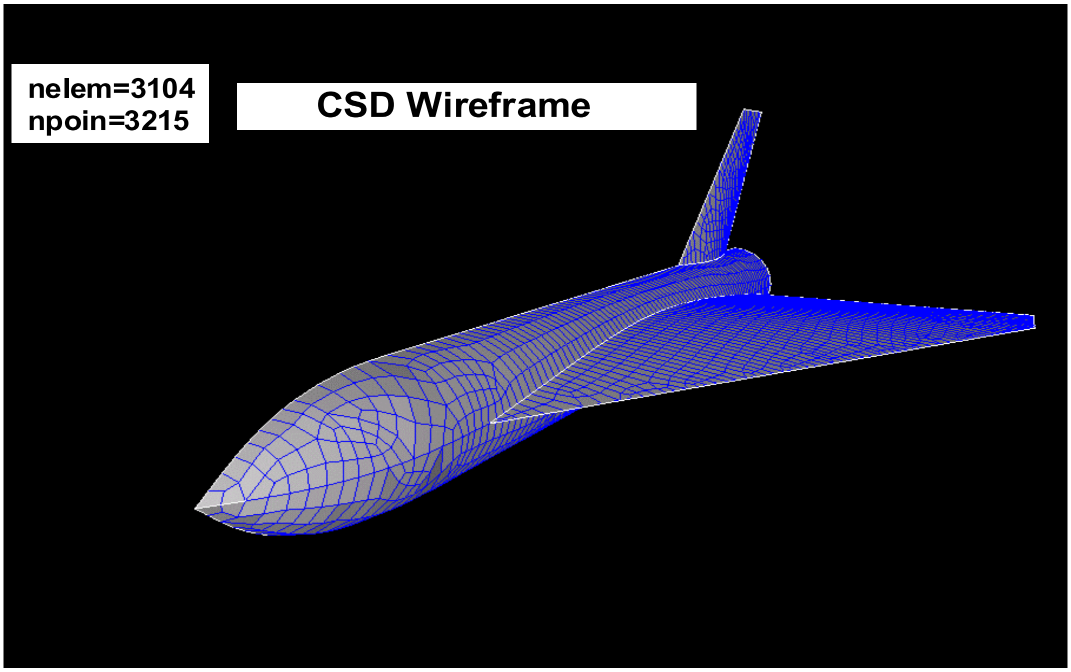


Fig. 2.1-3 CSD Mesh for Generic Airplane

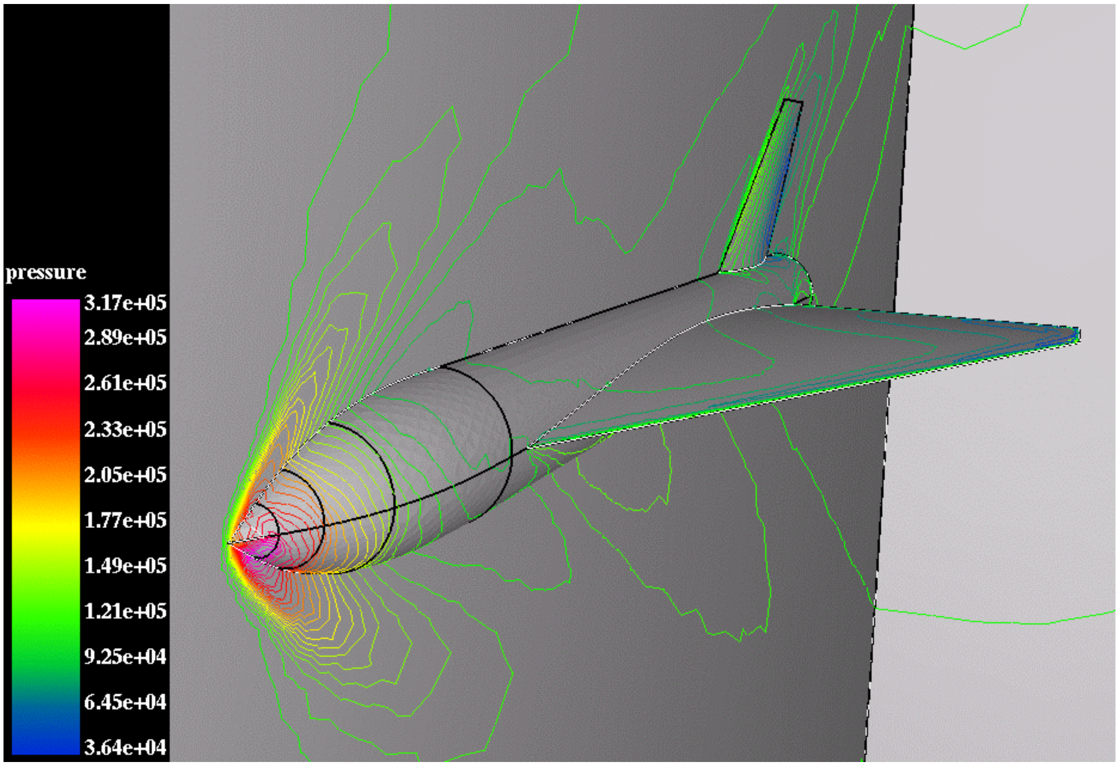


Fig. 2.1-4 Steady State Pressure Contours – Initial Iteration

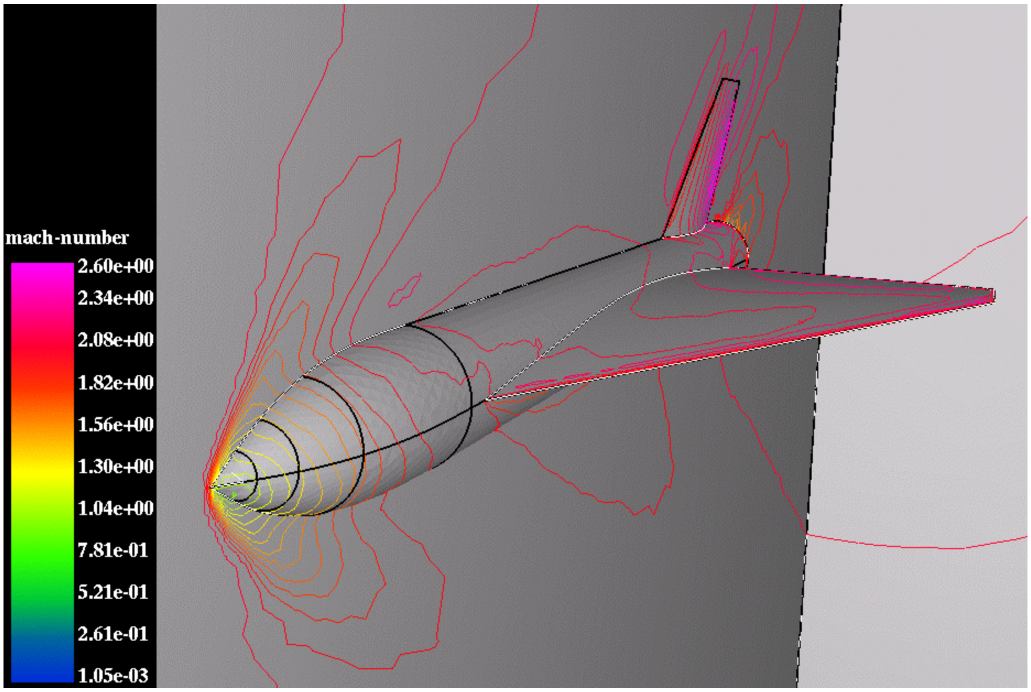


Fig. 2.1-5 Steady State Mach Contours – Initial Iteration

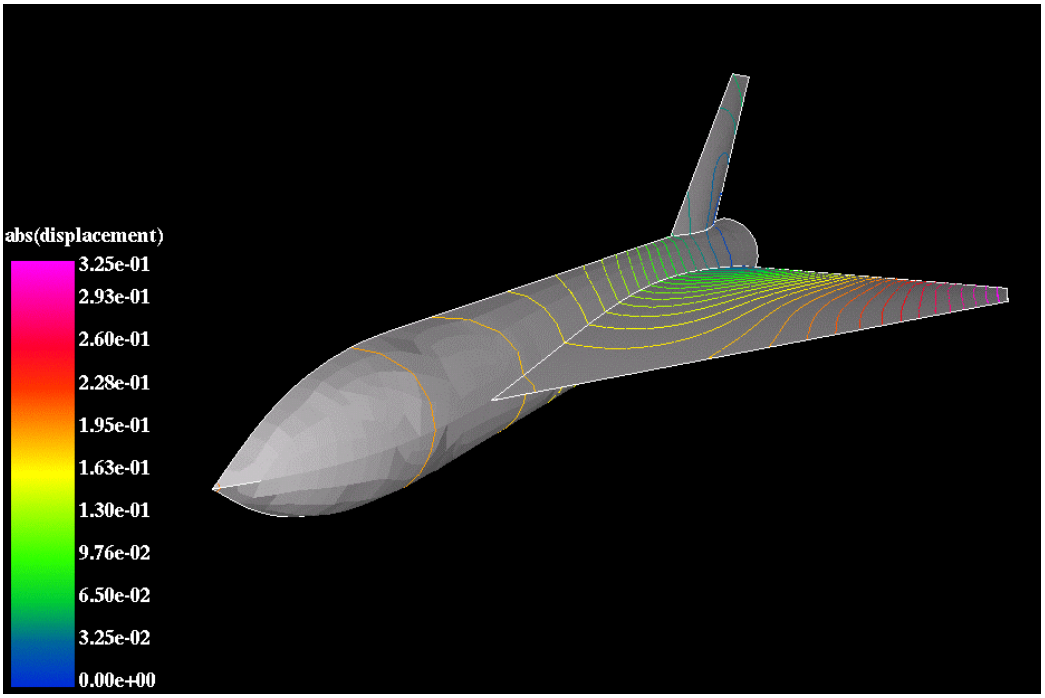


Fig. 2.1-6 CSD Deformation Contours – Initial Iteration

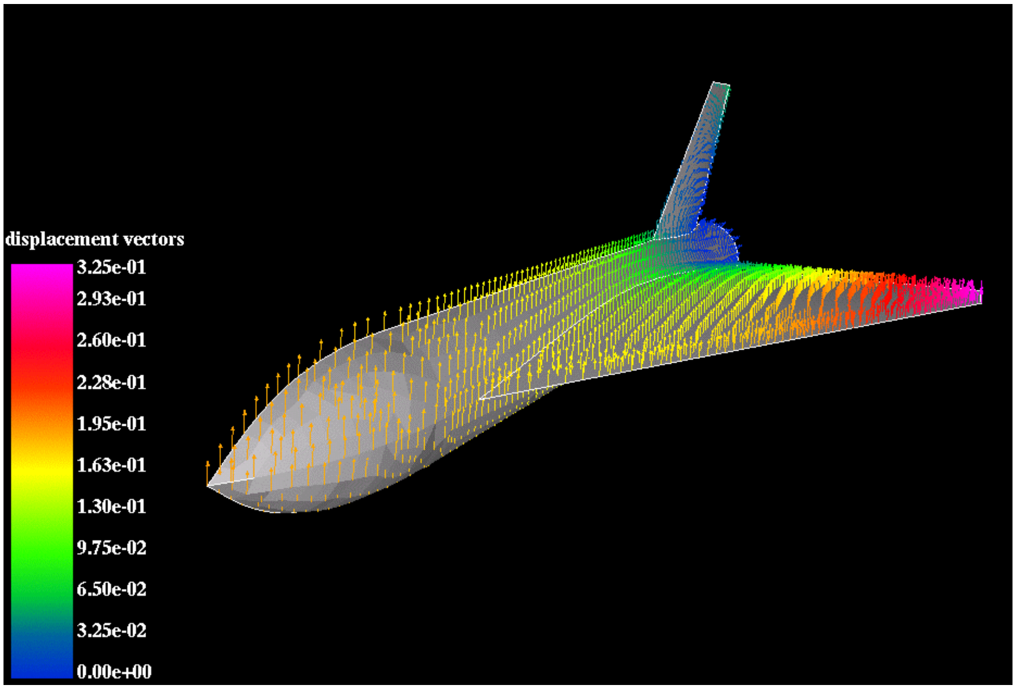


Fig. 2.1-7 CSD Surface Velocity – Initial Iteration

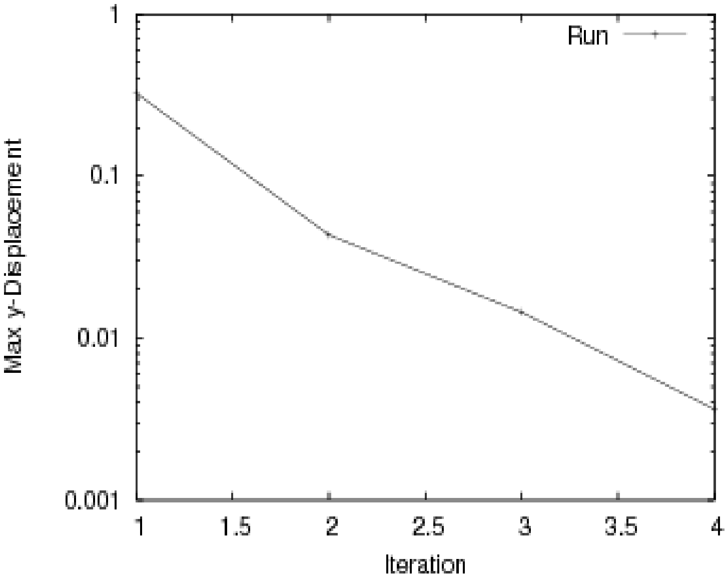


Fig. 2.1-8 Convergence History

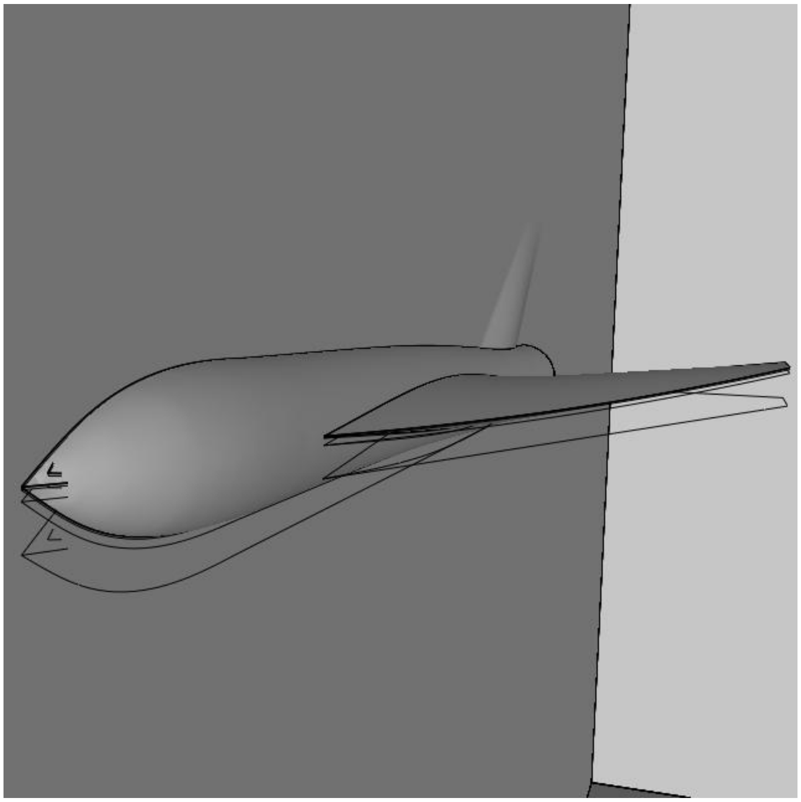
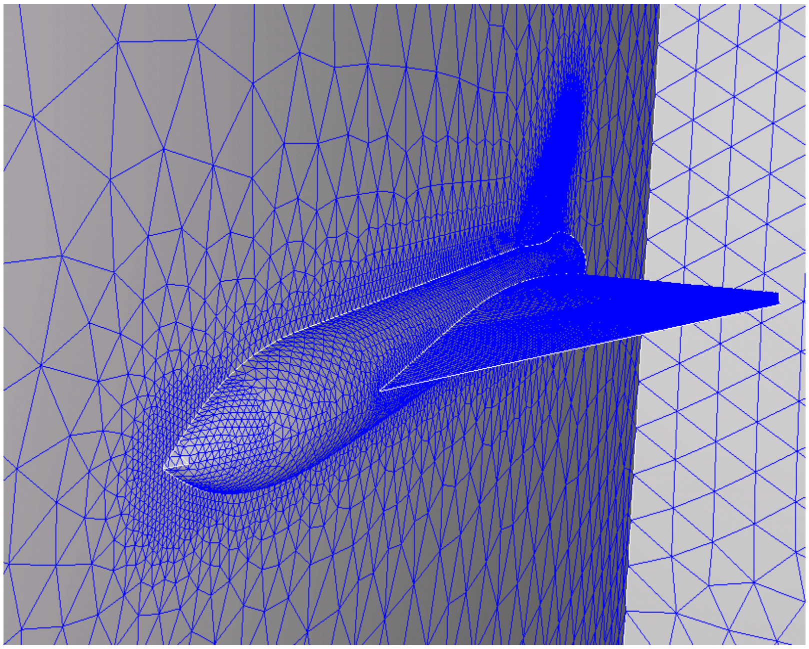
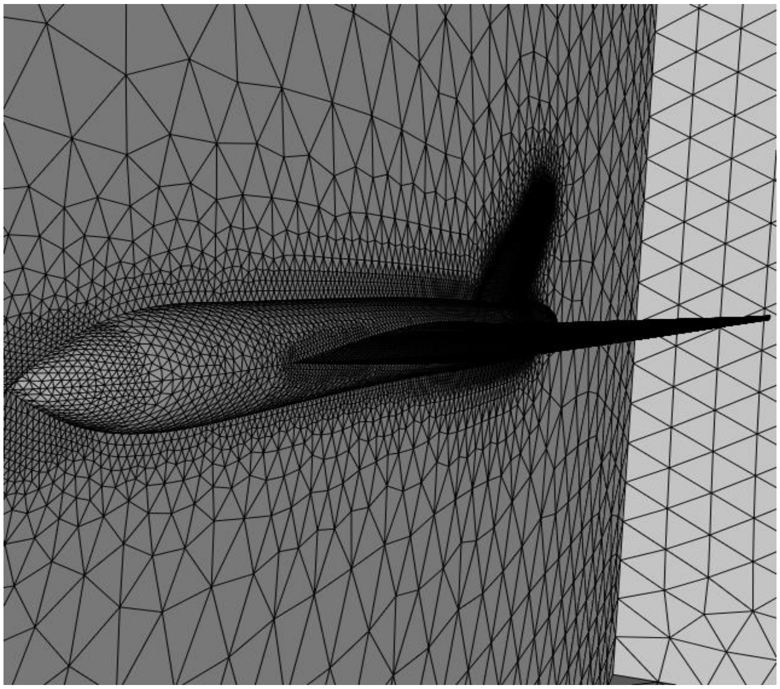


Fig. 2.1-9 Initial and Final Airplane Shape



a) Initial CFD Mesh



b) Final CFD Mesh

Fig. 2.1-10 Initial and Final CFD Meshes

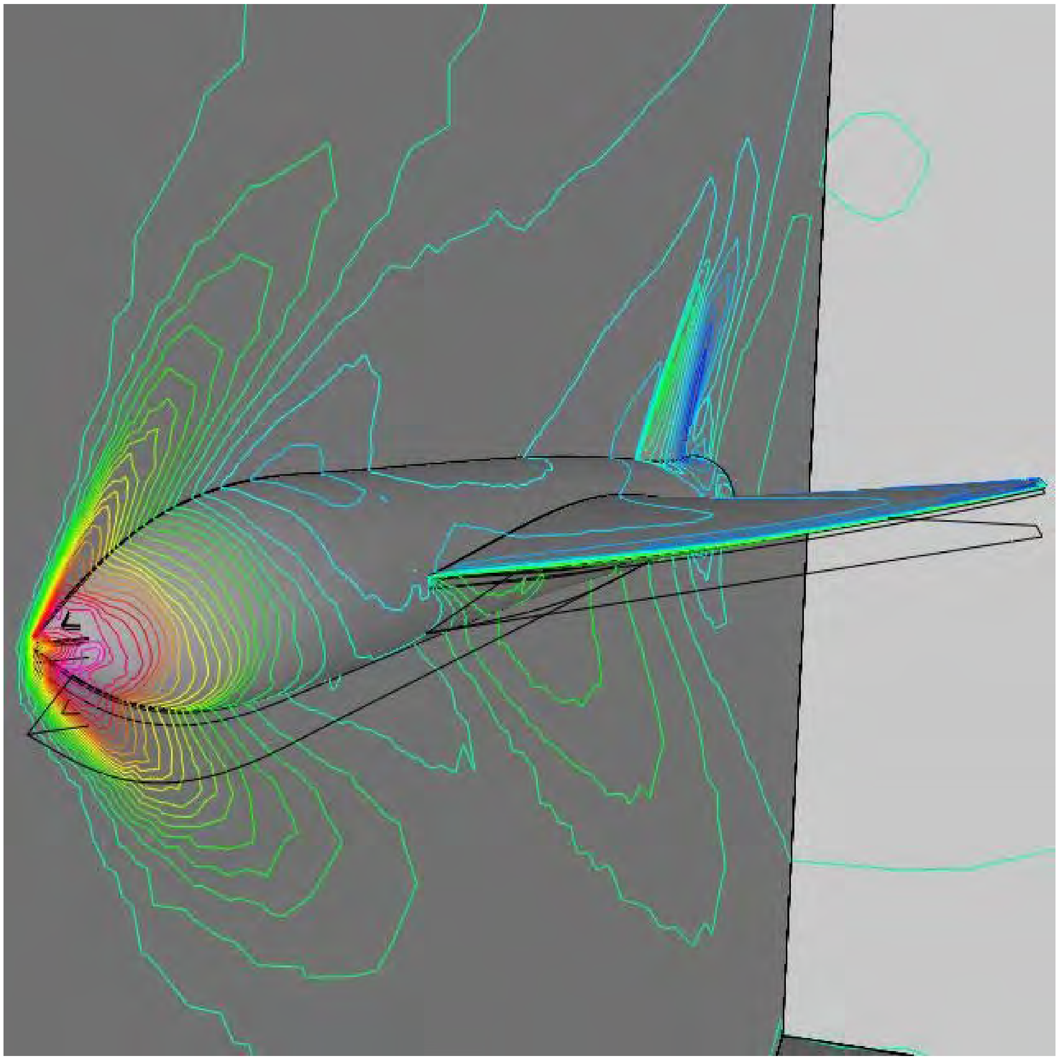


Fig. 2.1-11 Final Pressure Contours and Surface Positions

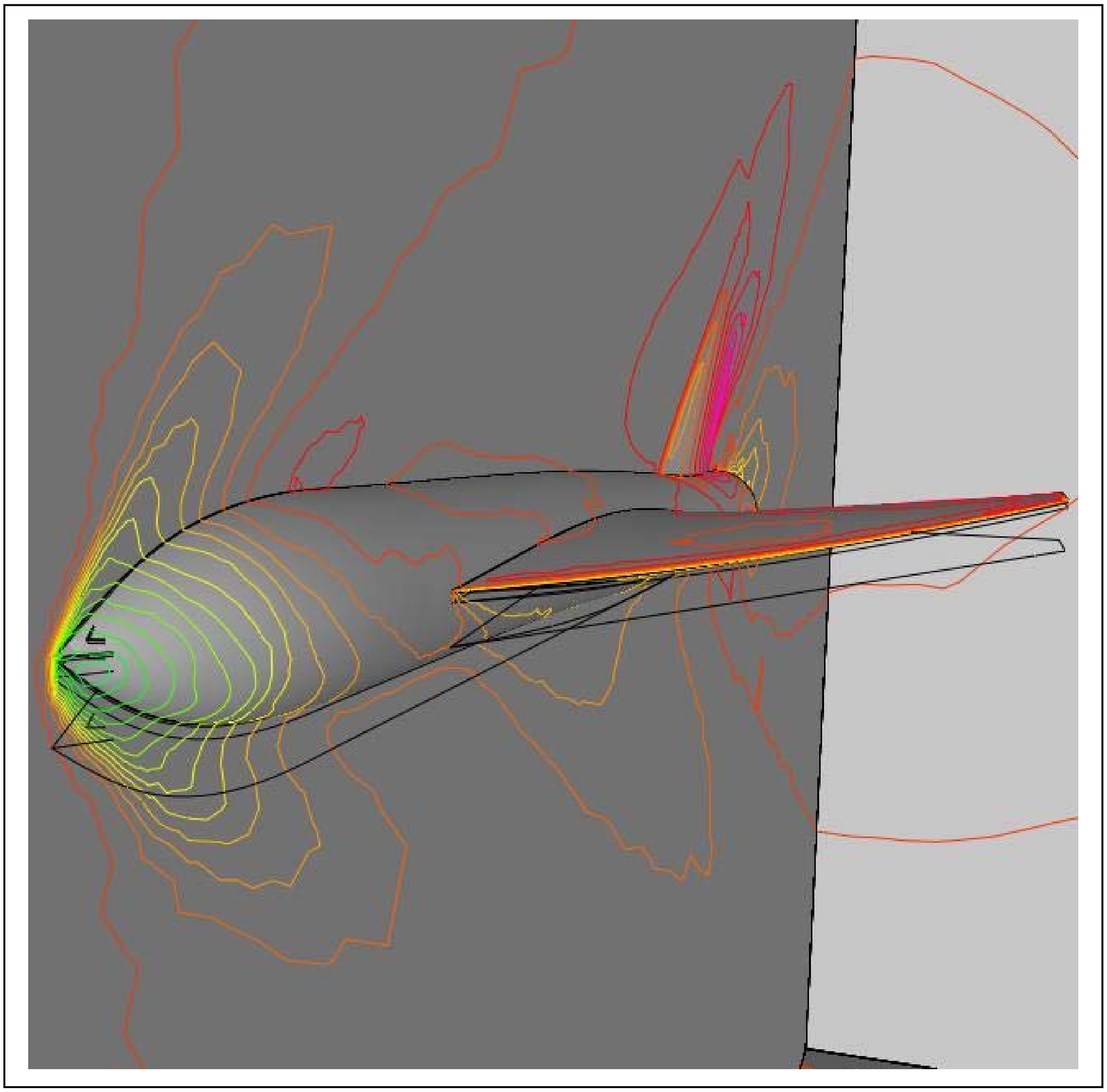


Fig. 2.1-12 Final Mach Contours and Surface Positions

2.2 Small Deformation Transient

For this transient case, the generic airplane configuration was filled-in with internal structure. The wing and the fuselage now contain internal beams which will take some of the pressure loadings. Fig. 2.2-0 shows some of this added structure.

The unsteady solution is shown here as plots of surface velocity vectors (colored on absolute value of velocity) and pressure contours. The pictures are shown for 8 time points in the transient run.

These are shown as Figs. 2.2-1 through Fig. 2.2-8. An animation of this solution shows this fluid – structure interaction more dramatically.

An enlarged view of the mesh velocity pictures are given in Figs. 2.2-9 through 2.2-12 for 4 time points. The enlargement near the tail shows the movement of the airplane wing.

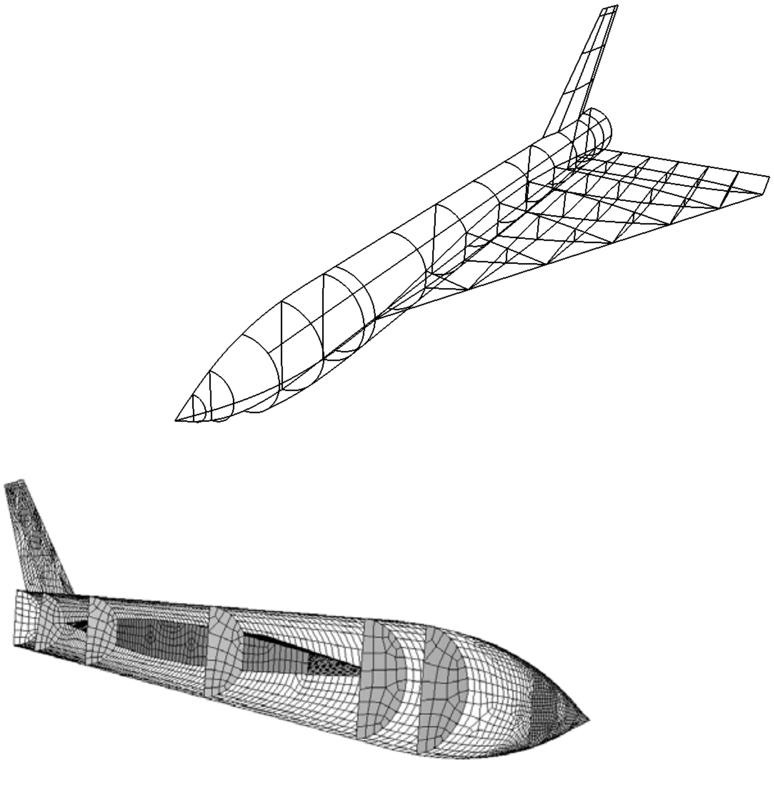
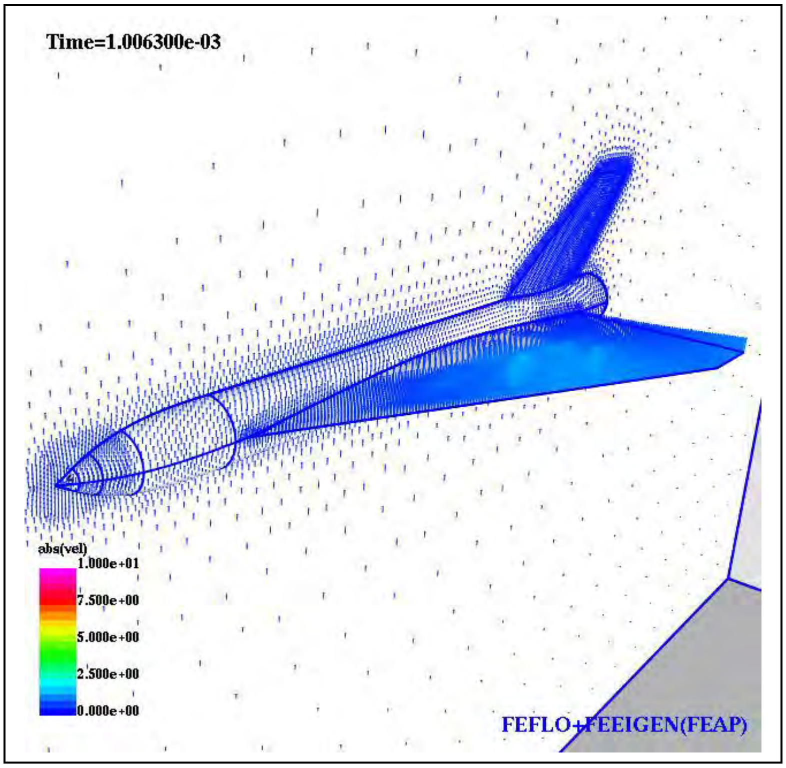


Fig. 2.2-0 Generic Airplane with Internal Structure Added



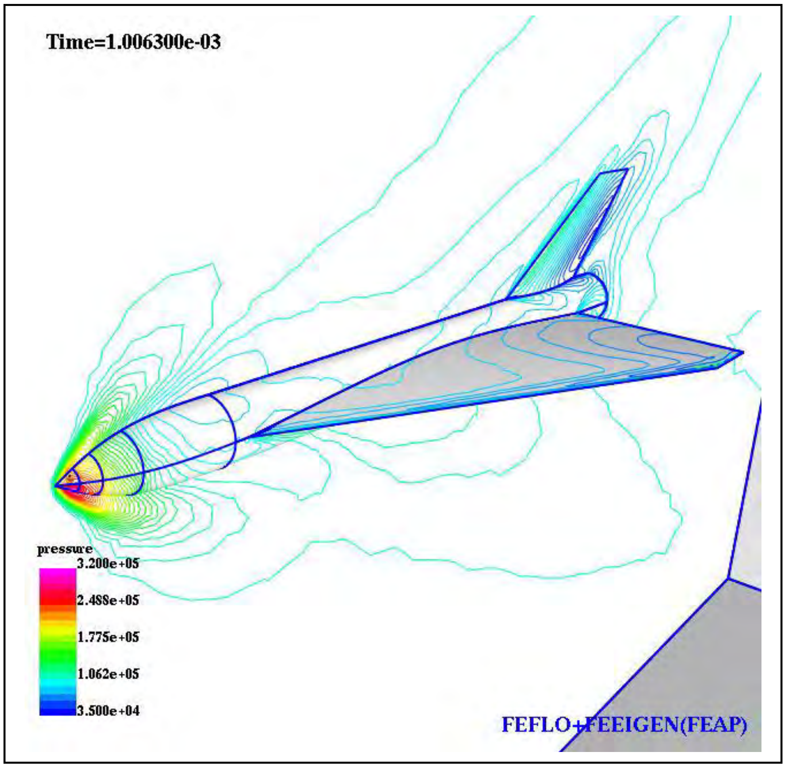
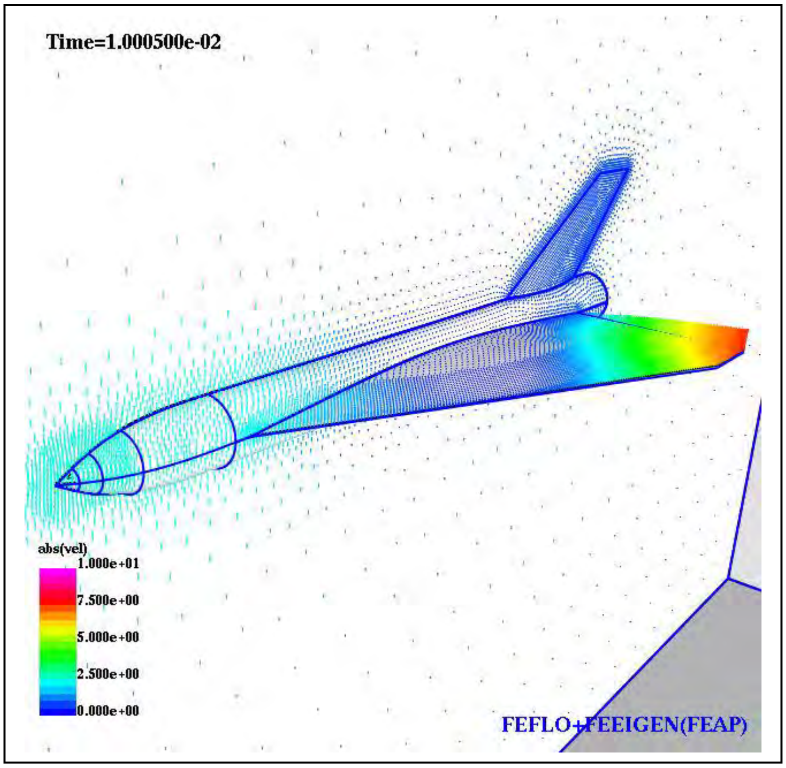


Fig. 2.2-1 Mesh Velocity and Pressure Contours at Time Point # 1



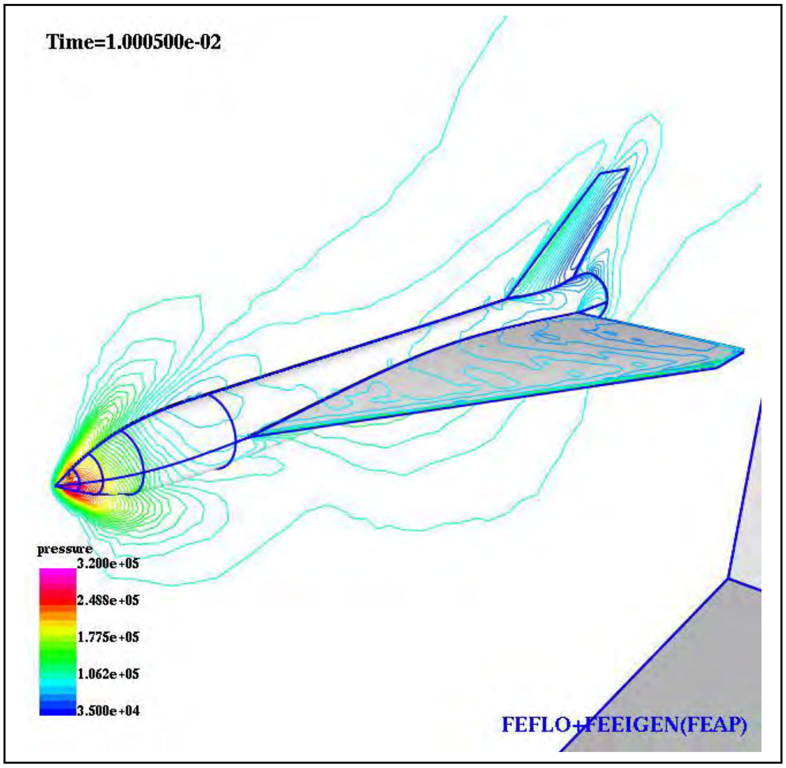
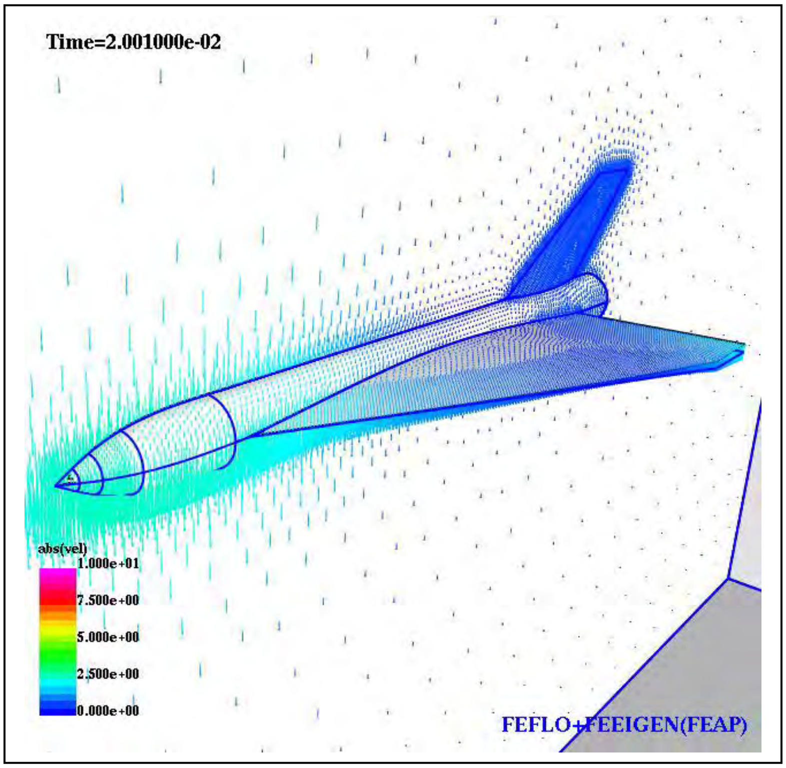


Fig. 2.2-2 Mesh Velocity and Pressure Contours at Time Point # 2



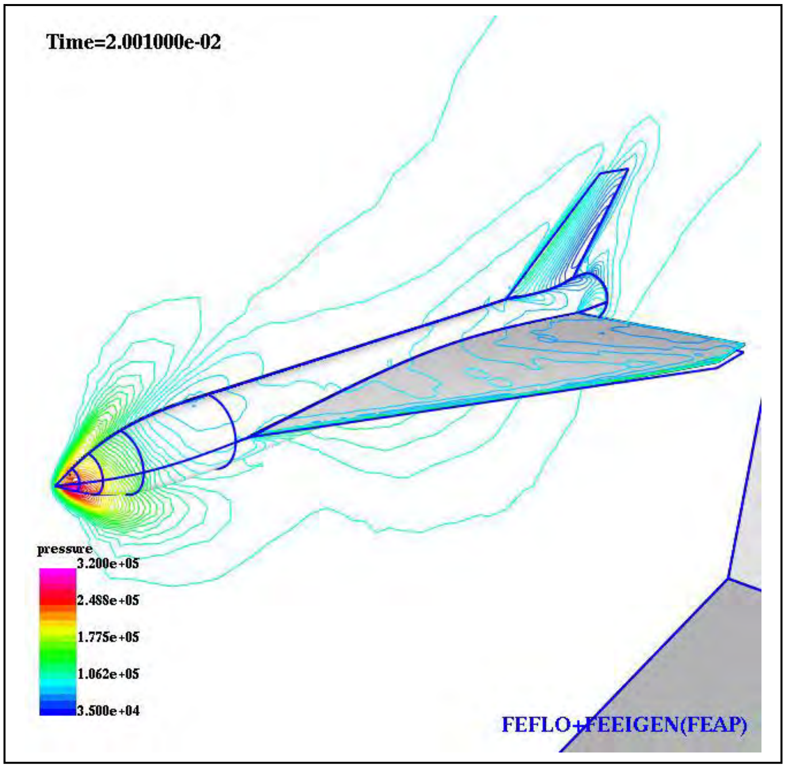
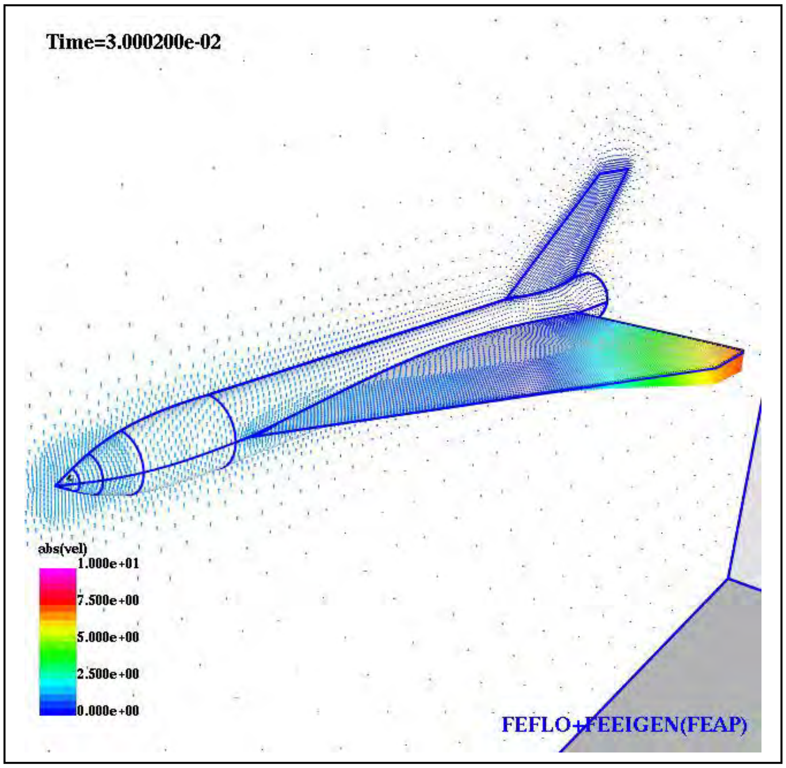


Fig. 2.2-3 Mesh Velocity and Pressure Contours at Time Point # 3



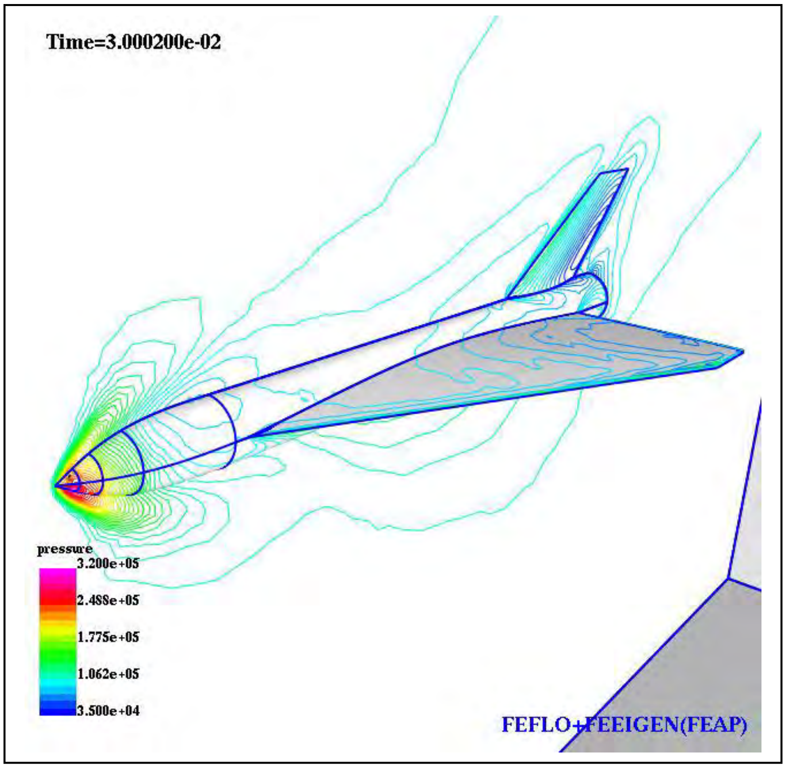
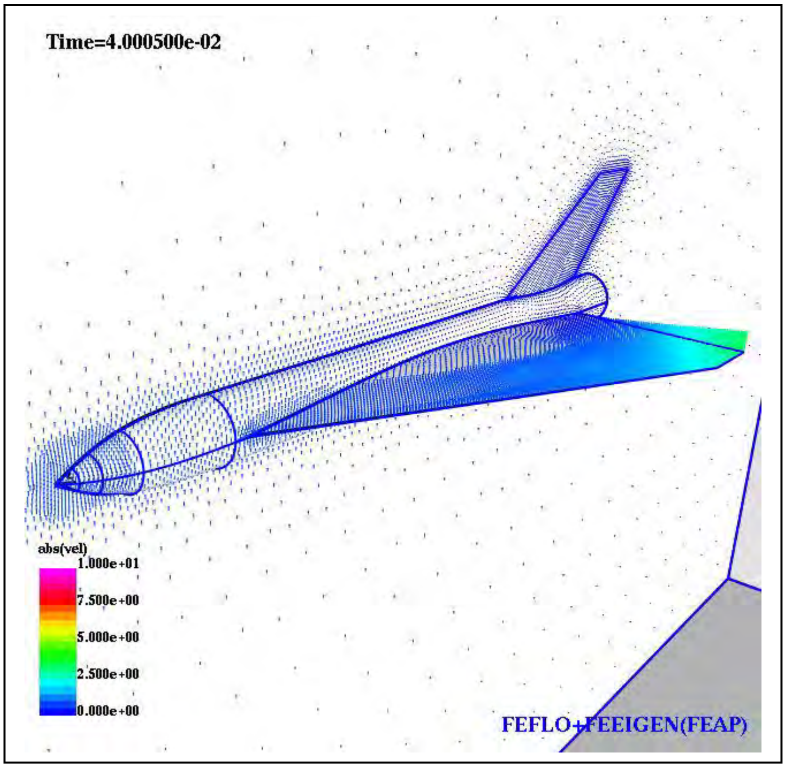


Fig. 2.2-4 Mesh Velocity and Pressure Contours at Time Point # 4



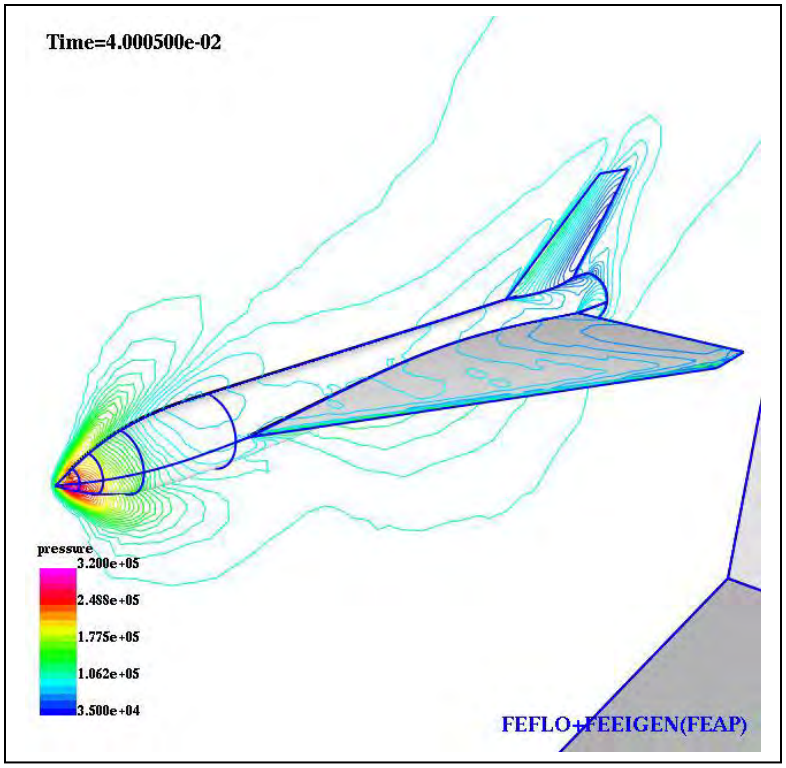
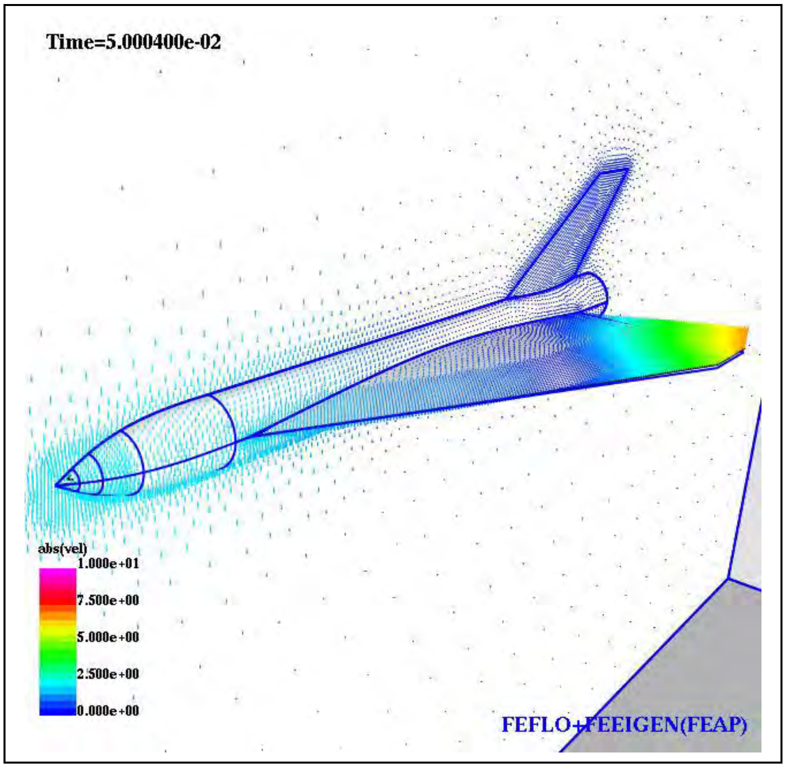


Fig. 2.2-5 Mesh Velocity and Pressure Contours at Time Point # 5



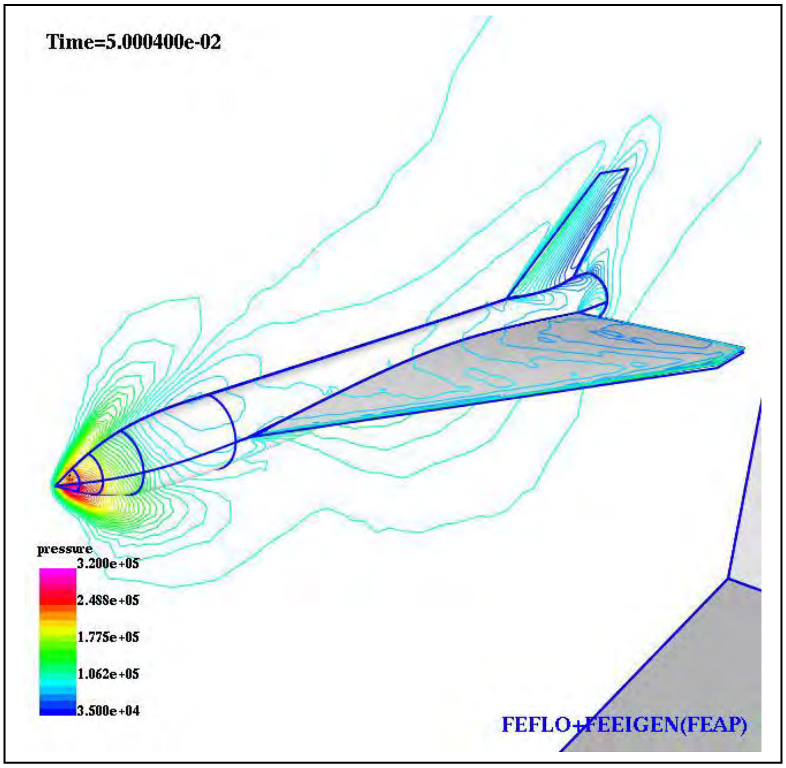
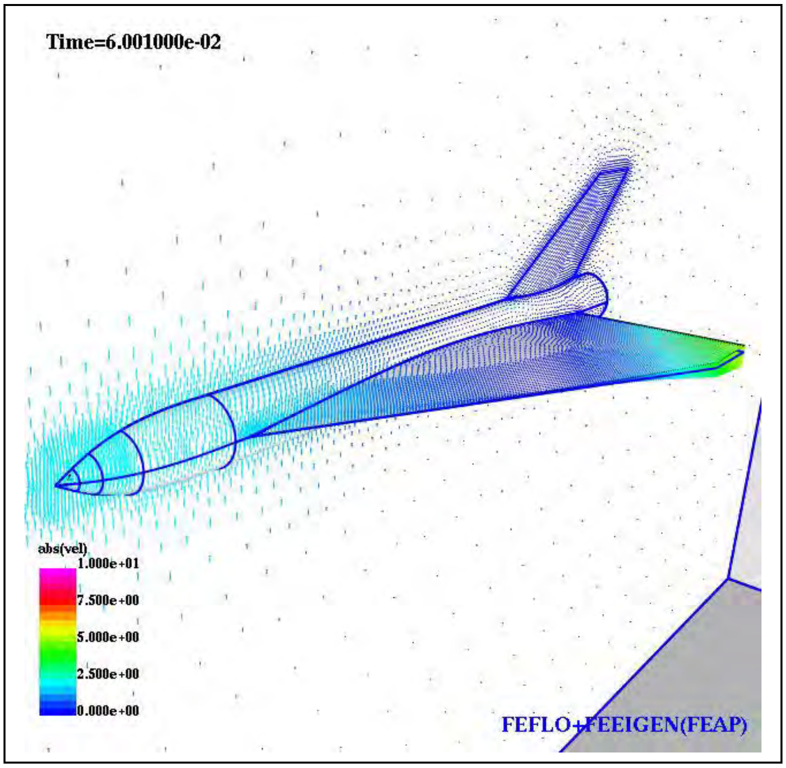


Fig. 2.2-6 Mesh Velocity and Pressure Contours at Time Point # 6



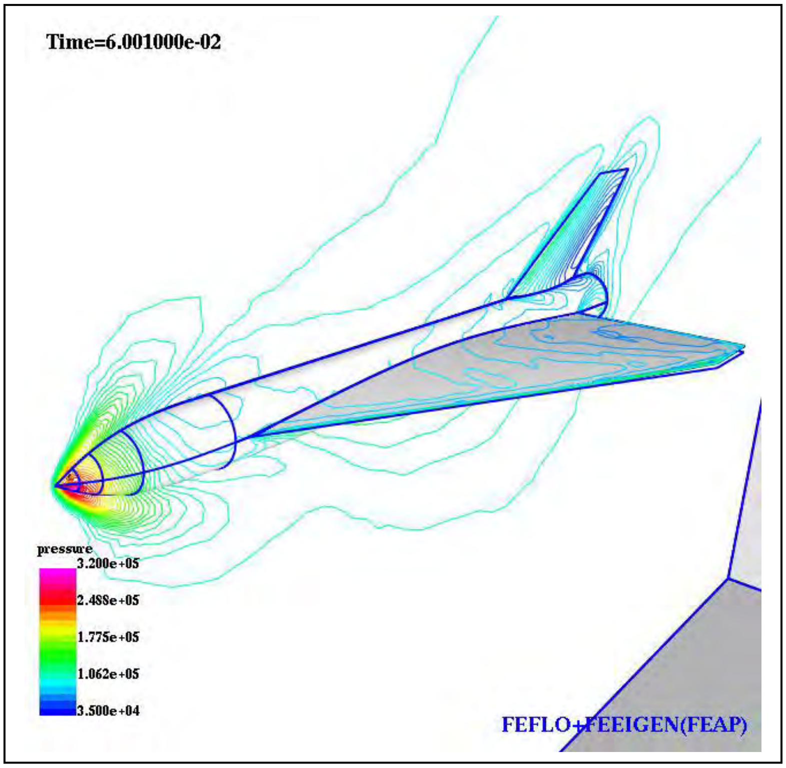
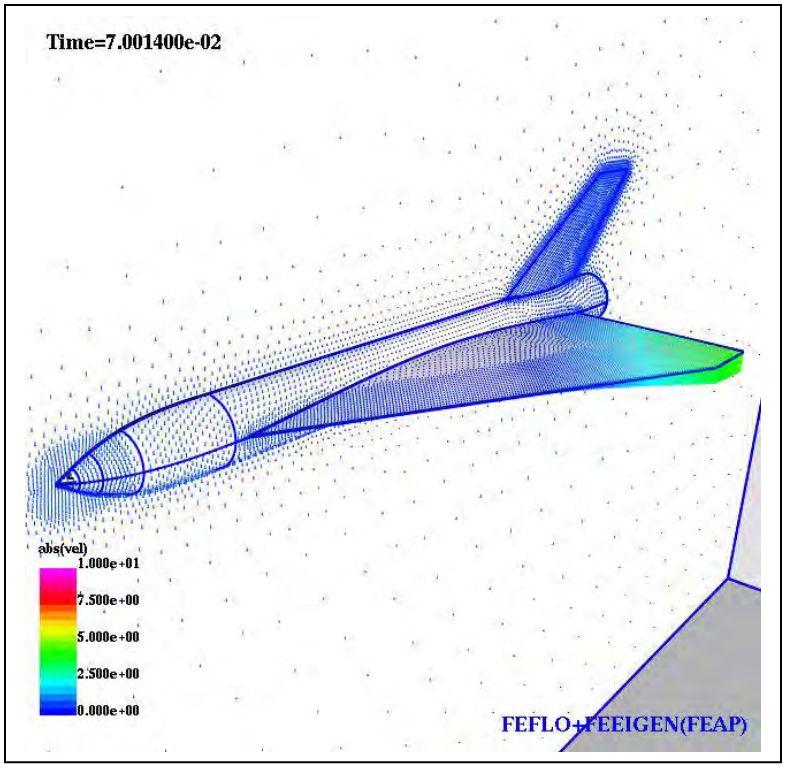


Fig. 2.2-7 Mesh Velocity and Pressure Contours at Time Point # 7



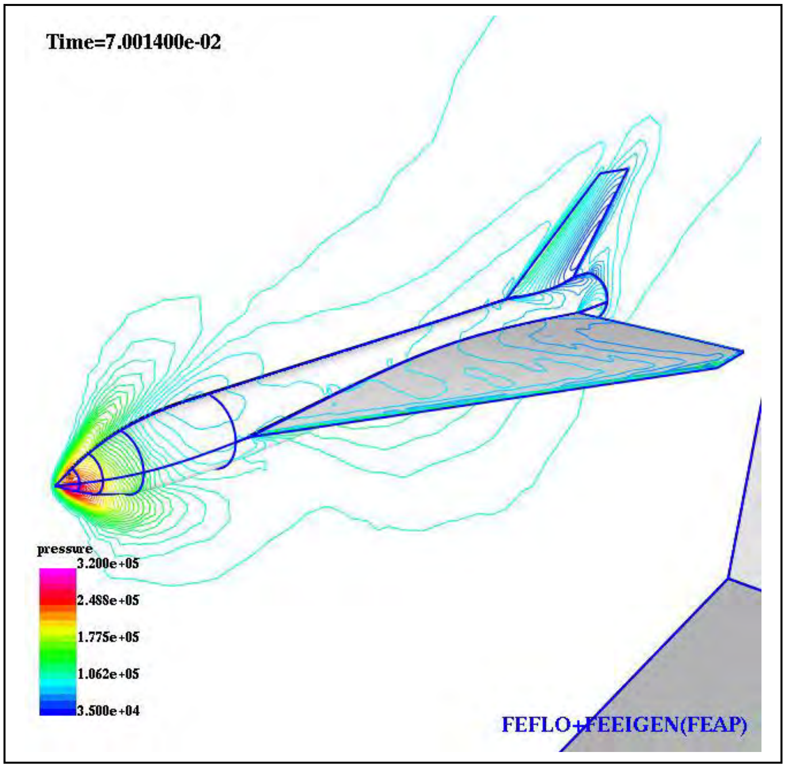


Fig. 2.2-8 Mesh Velocity and Pressure Contours at Time Point # 8

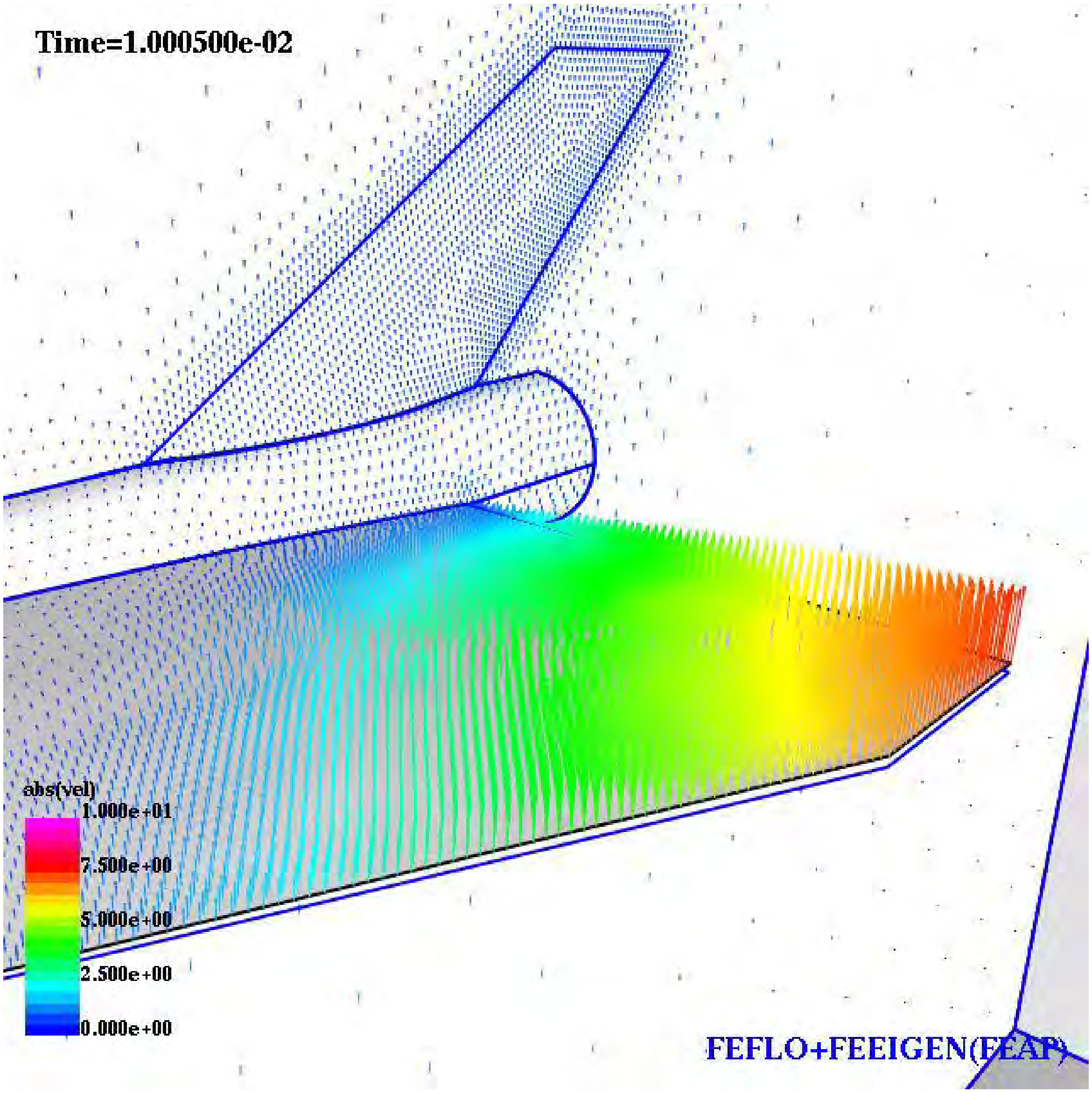


Fig. 2.2-9 Mesh Velocity Enlarged at Time Point # 2

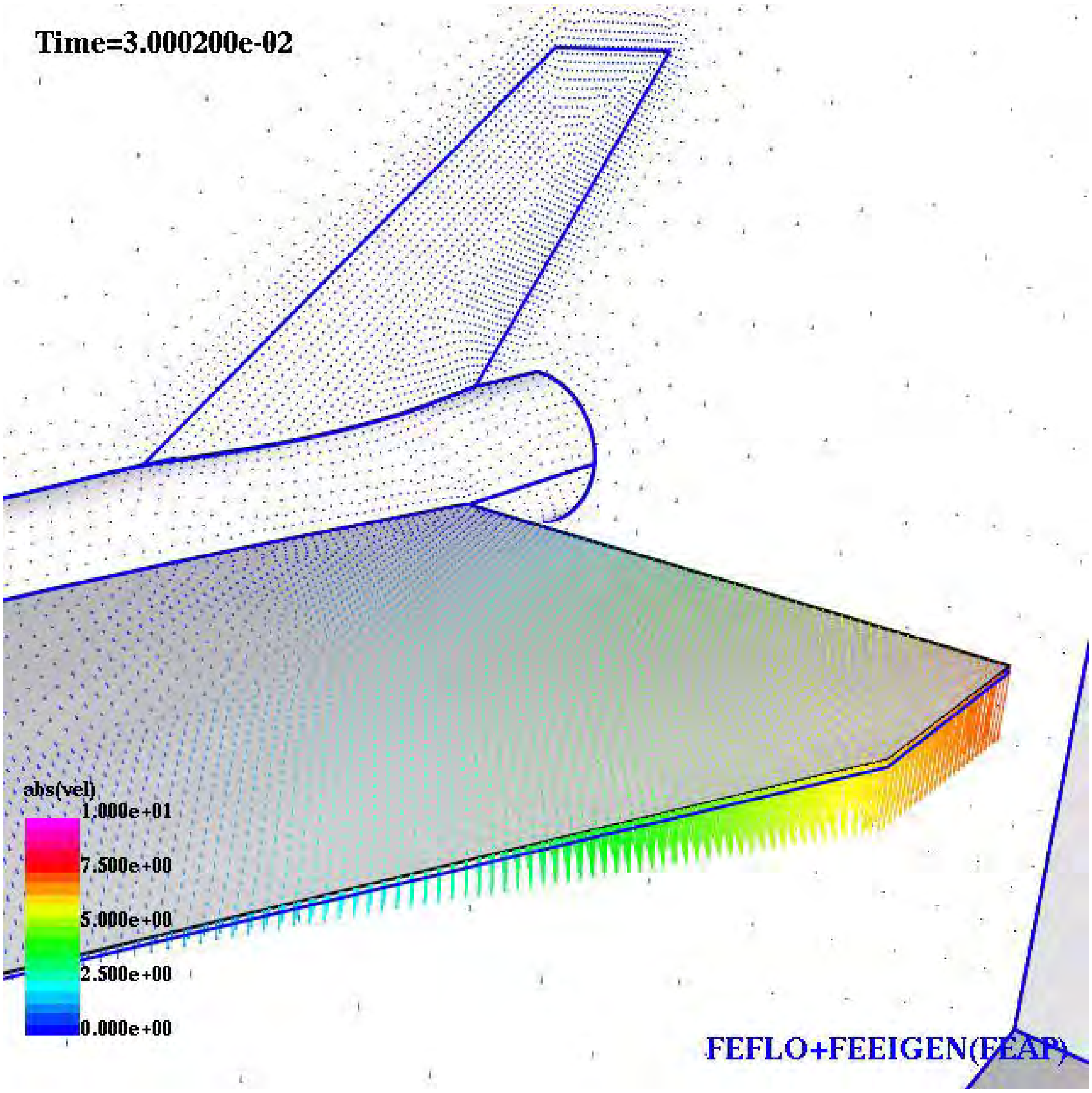


Fig. 2.2-10 Mesh Velocity Enlarged at Time Point # 4

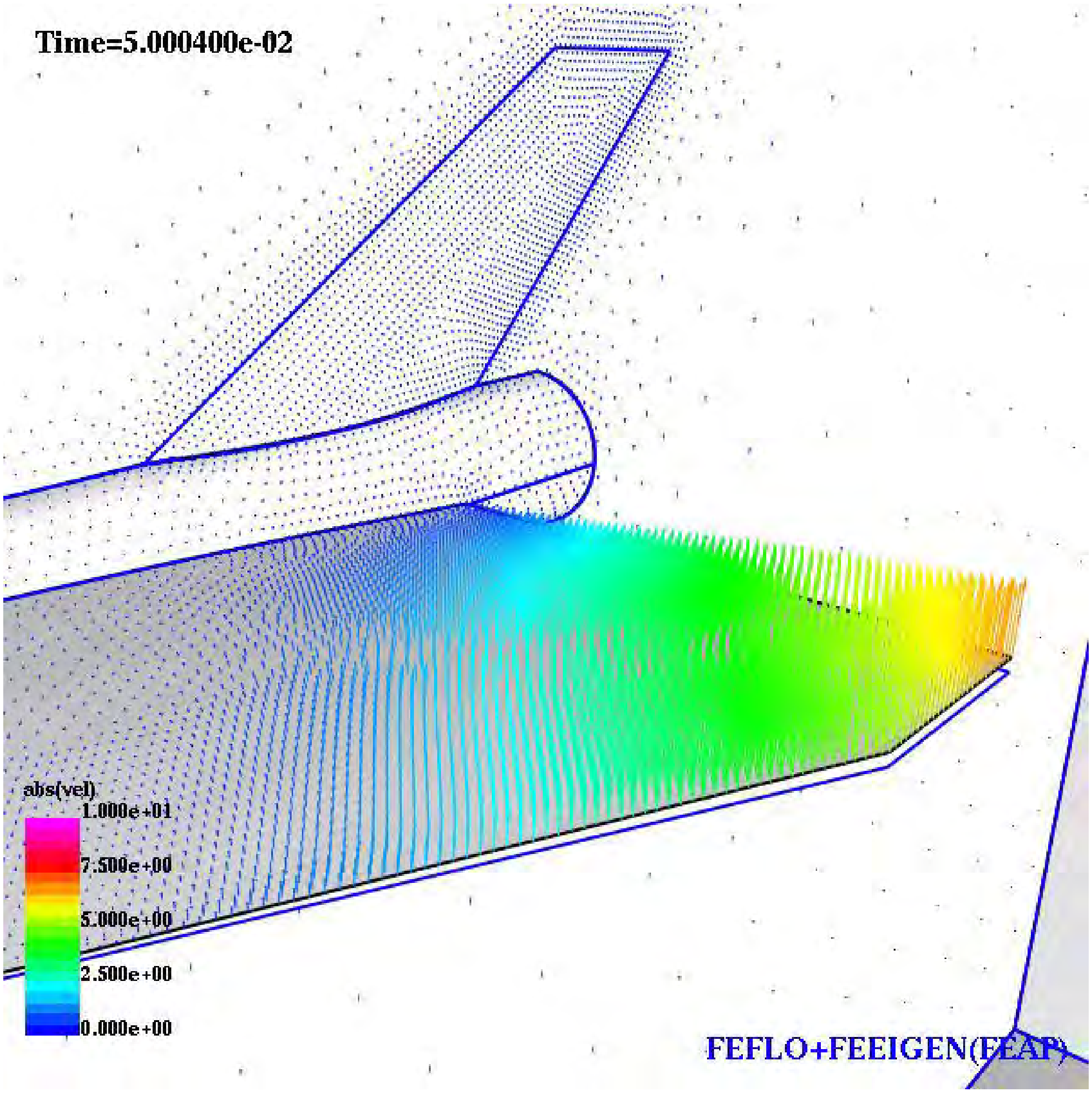


Fig. 2.2-11 Mesh Velocity Enlarged at Time Point # 6

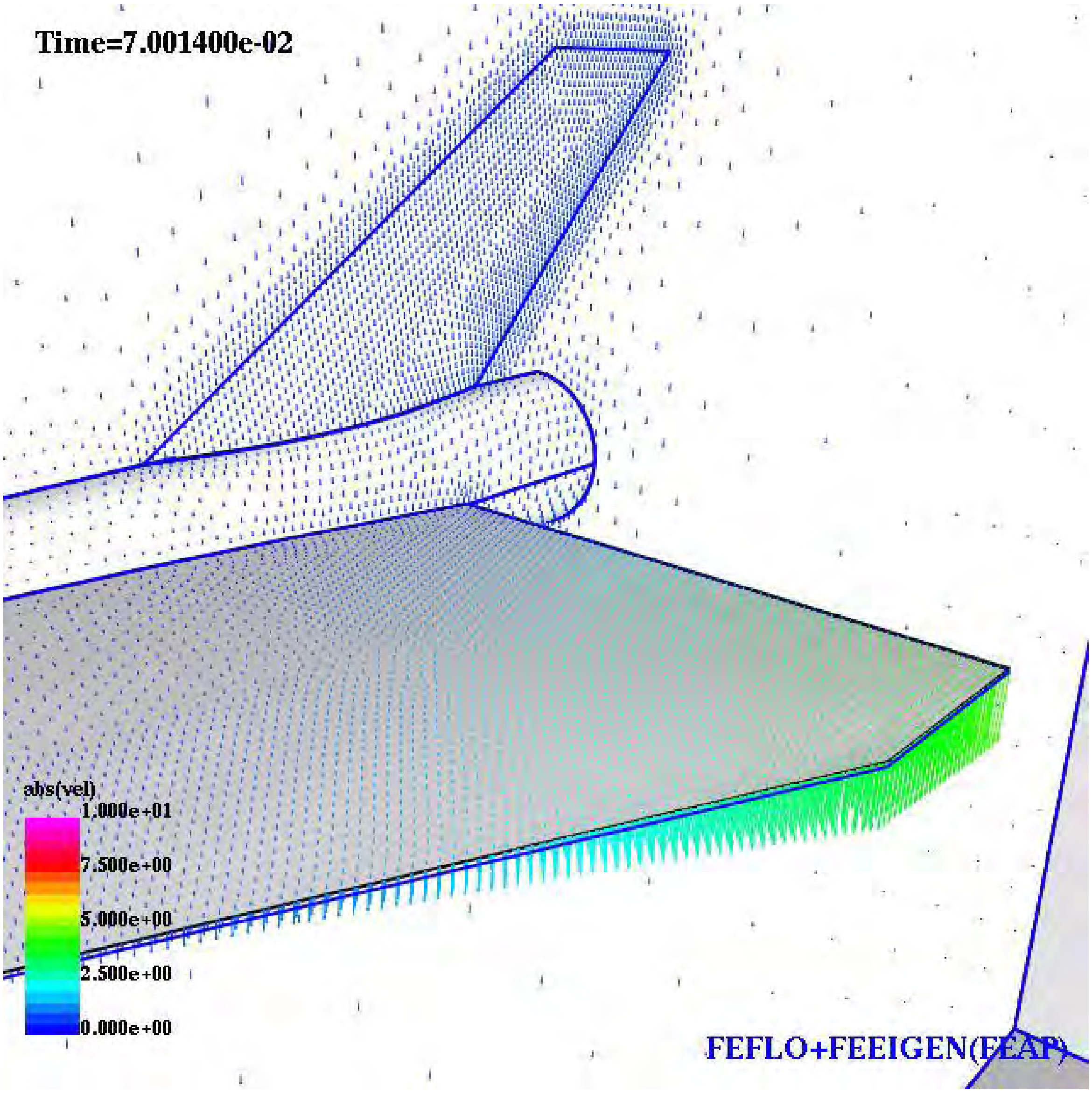


Fig. 2.2-12 Mesh Velocity Enlarged at Time Point # 8

2.3 Large Deformation Steady State

In the previous sections, we had linked the FEFLO CFD code to the linear FEAP module for elastic, small deformation cases. In this section, we present linkage to the large deformation module of FEAP for plastic strains. This is important if one is trying to assess the ultimate load bearing capability (or maximum velocity/dynamic head) of a vehicle.

The FEAP manual was consulted and several smaller test cases which came from Berkeley with the code were generated and run. The results showed that for a modest number of elements going into plastic deformation, FEAP obtains the results very quickly. Typically only 3-4 Newton iterations are required for convergence of the non-linear equations.

In order to link FEAP via FEEIGEN to FEFLO, a conversion from FEAP output to FEEIGEN-readable input is required. We developed such a converter, and are now able to automate the coupling process completely.

In order to ascertain in a more quantitative way the results obtained from FEAP, we wrote input modules to the FEPOST post-processor so that displacements and stresses obtained by FEAP could be displayed. At present, displacement and grid connectivity can be read for shell and beam elements. Stresses at the Gauss-points can be read for shell and beam elements, but are displayed as points. The intent is to extend this read capability to include stress recovery at points (for contour line/shading plots), as well as other element types

The plastic deformation option was then applied to the detailed generic flyer used previously for the small deformation case. The wireframe is shown in Fig. 2.3-1. The top shows the external structure which is modeled with shell elements. Some internal structure was added to the current case. These are shown the bottom of the figure. This model has internal structures in the wing, tail and body.

Several materials were specified in order to allow for shells and beams of different cross-sectional areas. The final mesh, composed from beam and shell elements, was generated automatically with FEGEN and outputted in FEAP format. The meshes are shown in Fig. 2.3-2 with the FEFLO CFD mesh on the top and the FEAP mesh in the bottom figure.

The steady-state coupled solution is given in Figs. 2.3-3 through 2.3-6. The displacement contours and displacement vectors are given in Fig. 2.3-3. These displacements are seen to be relatively large (~40 %). These same displacements are shown as wire frames in Fig. 2.3-4 comparing the original wireframe to the final shape (shaded). The top part of Fig. 2.3-5 shows the stresses plotted at the cell centers colored on stress level. The bottom figure is an exploded view on top of the wing. The displacement vectors are overlaid on the surface element stress plots. The final CFD flow field is given in Fig. 2.3-6. Pressure contours from a top and bottom view are given.

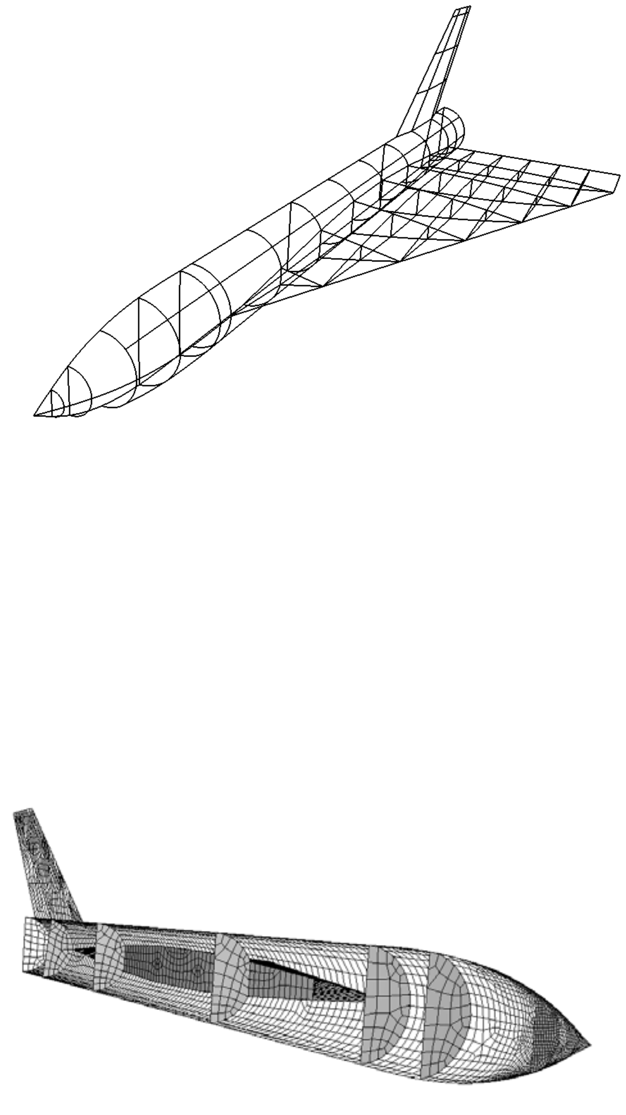


Fig. 2.3-1 Wireframe of Generic Airplane

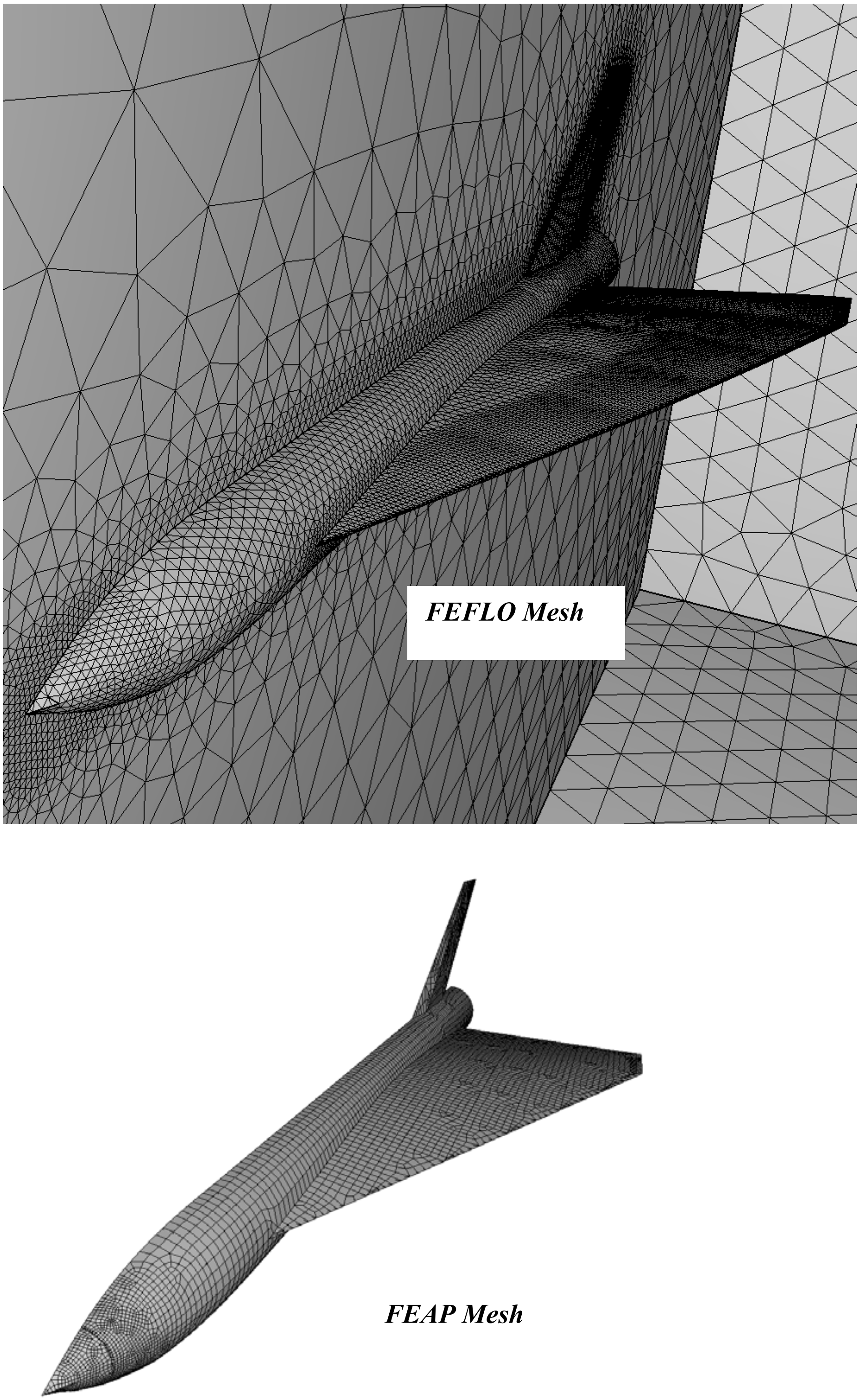


Fig. 2.3-2 CFD and CSD Mesh for Non-Linear Case

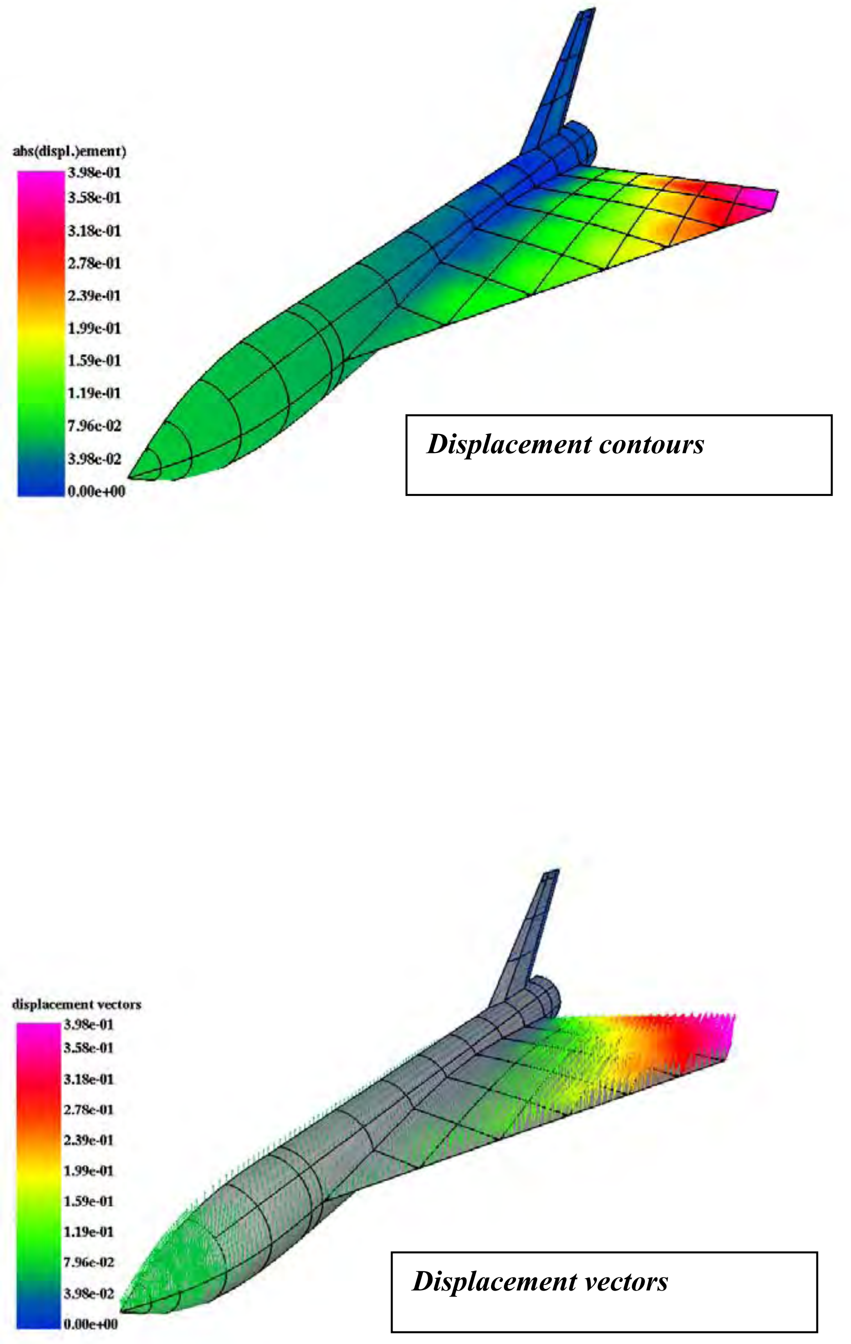


Fig. 2.3-3 Displacement Contours (top) and Vectors (bottom)

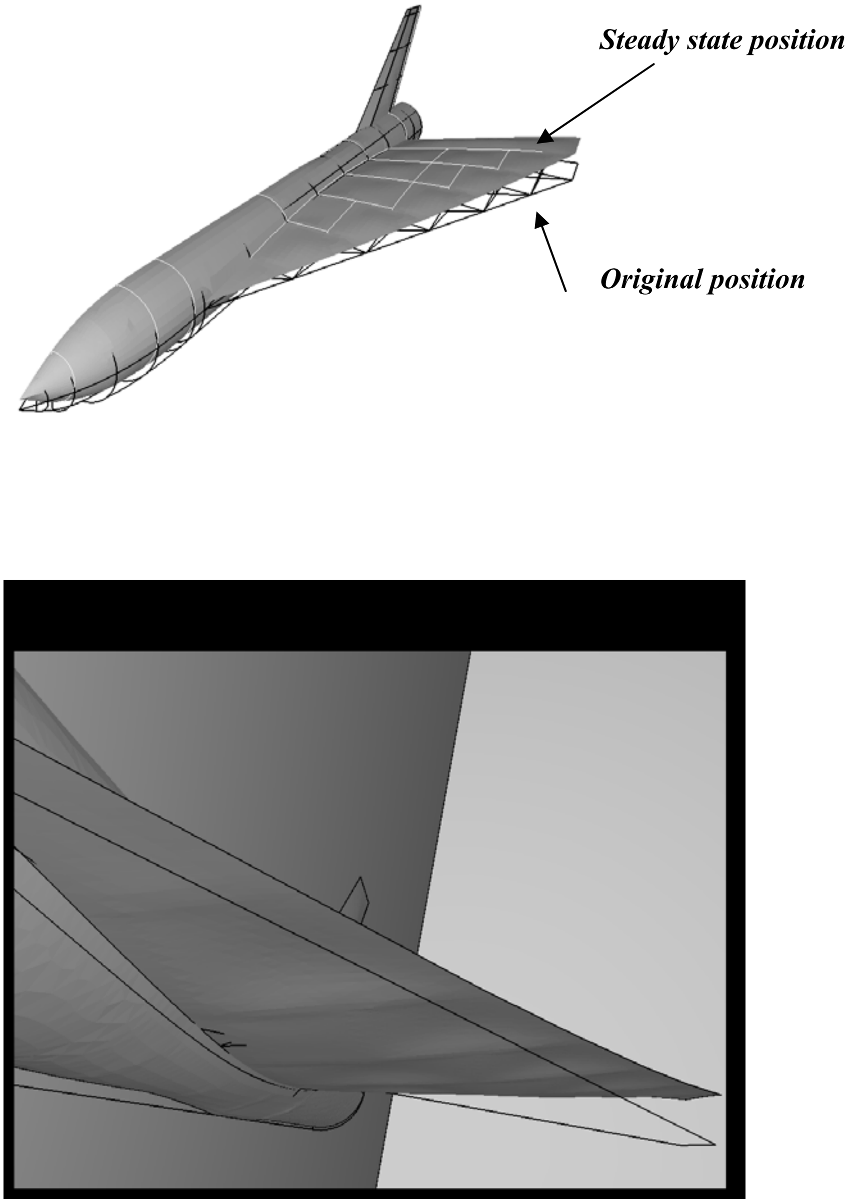
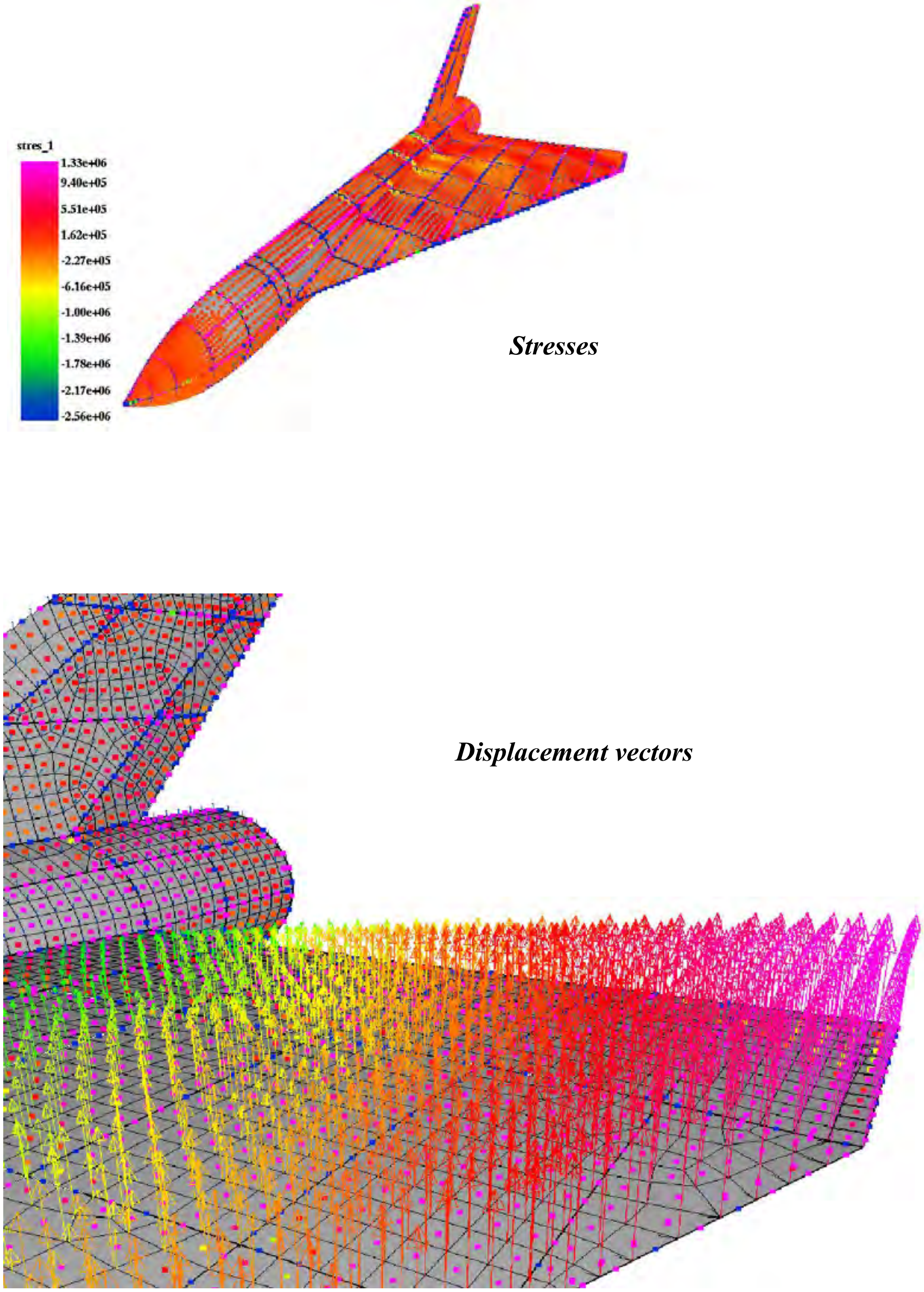


Fig. 2.3-4 Displacement Shown as Original (wire) and Final (solid)



Stresses

Displacement vectors

Fig. 2.3-5 Stresses at Cell Centers (top) and Displacement Vectors Overlaid with Node Stresses on Wing

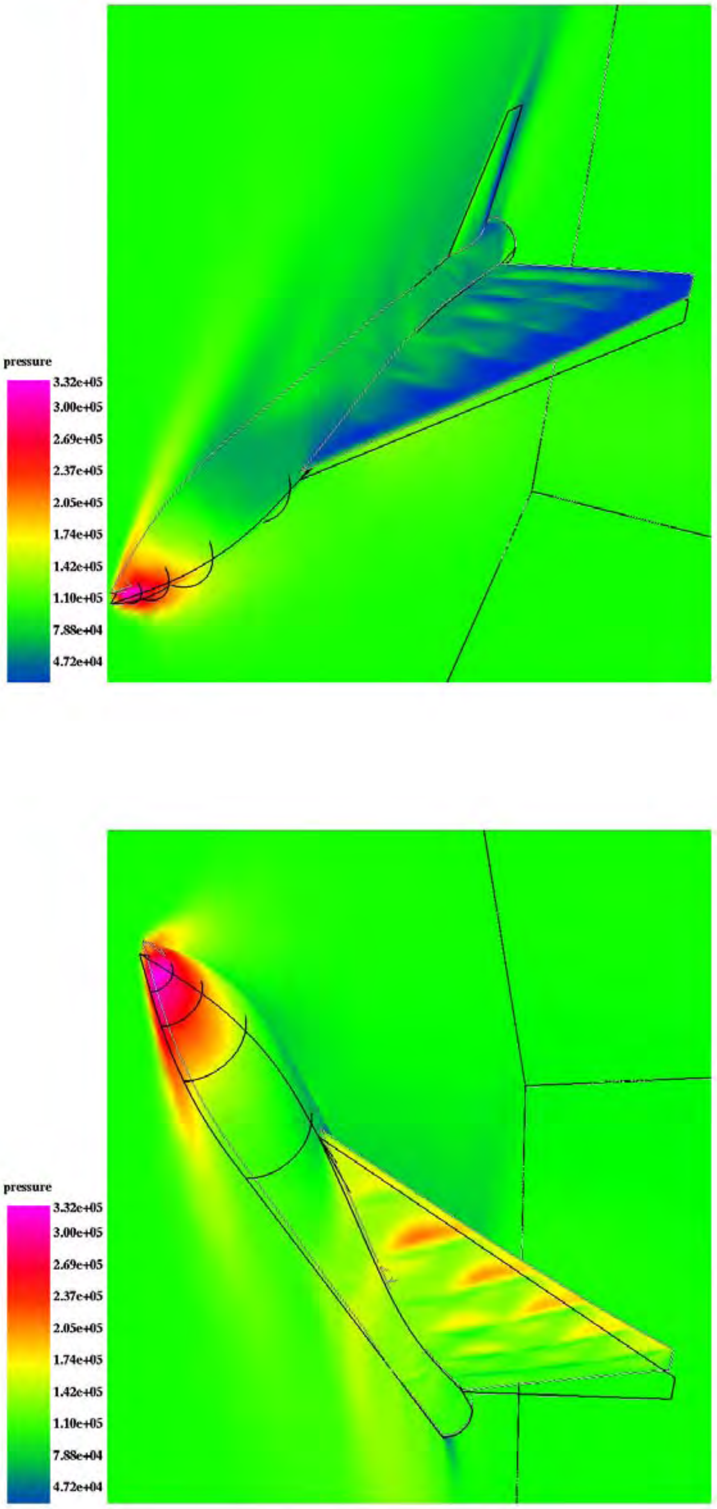


Fig. 2.3-6 Pressure Contours on Final Configuration (2 views)

3.0 EULER PLUS BOUNDARY LAYER INTEGRATION

3.1 FEFLO to HUD Code Integration

Methodology

The analysis of viscous flow over hypersonic vehicles may be addressed through a variety of methodologies ranging from the relatively simple and efficient to the highly sophisticated but usually more computationally intensive and less stable. The selection of methodologies requires trade-offs between desired accuracy, computational cost and engineering effort.

The most sophisticated, and potentially most accurate, approach utilizes a RANS CFD analysis of the entire flowfield. This high fidelity approach, with suitable modeling of turbulence, predicts viscous effects (aero-heating and shear) over the vehicle surface, including the separation of boundary layers. RANS solutions, however, are computationally expensive and often unstable. Extremely fine grid spacing is required near surfaces to adequately resolve boundary layers. This leads to computational costs and computer size requirements significantly greater than that for corresponding inviscid (Euler) solutions.

An alternate approach to RANS analysis is needed for operation in a preliminary design environment. A combination of inviscid Euler flow codes coupled to efficient boundary layer codes has been used for decades as a practical design tool. In this way, the aerodynamic coefficients lift, and drag can be computed accurately and, in addition, viscous effects such as skin friction and heat transfer can be determined. This approach provides a viscous flow analysis capability over full vehicle configurations with an efficient and practical design tool. An adaptive mesh Euler flow solver, FEFLO, has been delivered to NASA Langley for computing the inviscid outer flow on complete hypersonic vehicle configurations. This task is to interface a standard, government-supported, boundary layer code, HUD, to this FEFLO Euler solver.

The HUD code [1] is a well known solver at Langley which was developed to compute viscous boundary layer effects over 2D (axisymmetric or planar) surfaces utilizing prescribed boundary layer edge conditions as inputs. Typically, the edge conditions are determined through simpler methodologies. The HUD solution proceeds by marching downstream in the direction of flow. This code has enjoyed considerable success for over 20 years due to its accuracy and reliability for the 2D problems which it was developed to address.

The procedure developed in this contract combines the 3D inviscid CFD capability of FEFLO to compute flowfields outside of the boundary layer with HUD to compute boundary layer viscous effects over vehicle surfaces. Previous work in this area has been reported in [2]. The inviscid FEFLO output is used to provide boundary layer edge conditions to HUD. This provides a reasonable merging of the accuracy of FEFLO for 3D flowfield computations and the accuracy and reliability of HUD for viscous computations. This requires that FEFLO be run first to provide inviscid flowfield data over the vehicle surface. The FEFLO data is then processed to compute flow lines over the surface with flow properties along these flow lines. These data lines are then input into HUD to compute viscous effects (heat transfer and shear) over the vehicle surface. The resulting viscous computations are based on an assumed 2D (axisymmetric or planar) character for flow along the flow lines.

A software suite, including a modified HUD code, was developed to automate the pre-processing of FEFLO output data into a set of streamline data for use in HUD, followed by repeated executions of HUD over the set of streamlines, and finally post-processing the HUD data into plot files. Modifications to HUD were required for coupling to FEFLO and thus the new solver is named FEHUD.

Verification Results

The above procedure has been implemented into the multi-disciplinary FEMAP system.

In addition, a classic method from the literature was coded for comparison and checkout [3]. The method is based on integration of pressure and skin friction on the surface of the body. For the skin-friction calculation, the van driest skin-friction model is used. Empirical base drag is also included to the drag calculations. Flow solutions are obtained by an unstructured three-dimensional Euler solver. Supersonic attached flows are the focus, rather than subsonic and transonic flow. The method cannot be applied to separated and vertical flows at a high angle of attack.

A flat plate test case was run first as shown in Fig. 3.1-1 at 100 meters per second. The results are shown in Fig. 3.1-2. On the left side is the absolute shear contours from FEHUD and on the right side is the approximate method solution [3]. More contours are bunched up near the leading edge for the FEHUD solution. This expected since the FEHUD case is an actual boundary layer solution and the approximate method is simply an equation. However, the results look very much the same.

Further comparison is given in Fig. 3.1-3. Near the leading edge, the agreement is within 1% and even near the trailing edge, the difference is less than 10%. This shows that our FEFLO / HUD interface is working properly. The HUD heat flux contours are given in Fig. 3.1-4.

We next ran this code interface on the generic HRRLS vehicle. We also run the HRRLS vehicle through the procedure of using the FEFLO Euler solution followed by the approximate method [3] to get shear. The mesh is shown in Fig. 3.1-5 with the Euler pressure contours shown in Fig. 3.1-6. The Abs Shear contours on the vehicle are plotted in Fig. 3.1-7. A close up of shear contours are shown in Fig. 3.1-8.

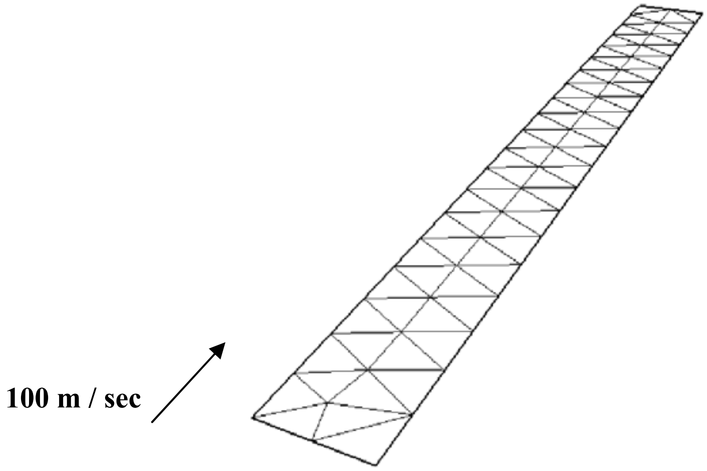


Fig. 3.1-1 Flat Plate for Test of FEFLO – FEHUD coupling

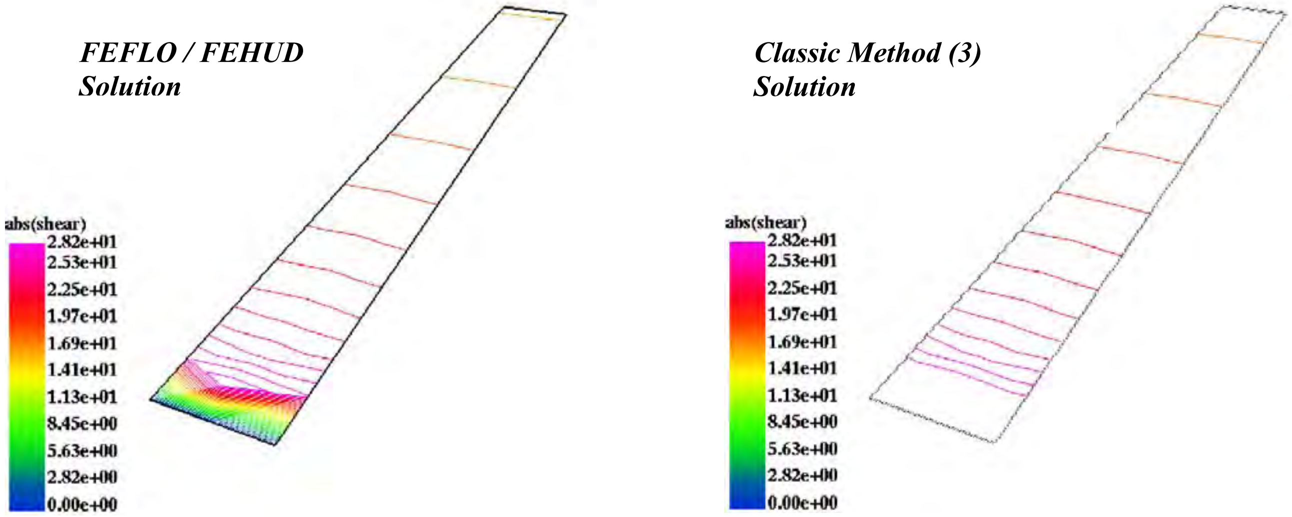


Fig. 3.1-2 Absolute Shear Contours for Flat Plate Case

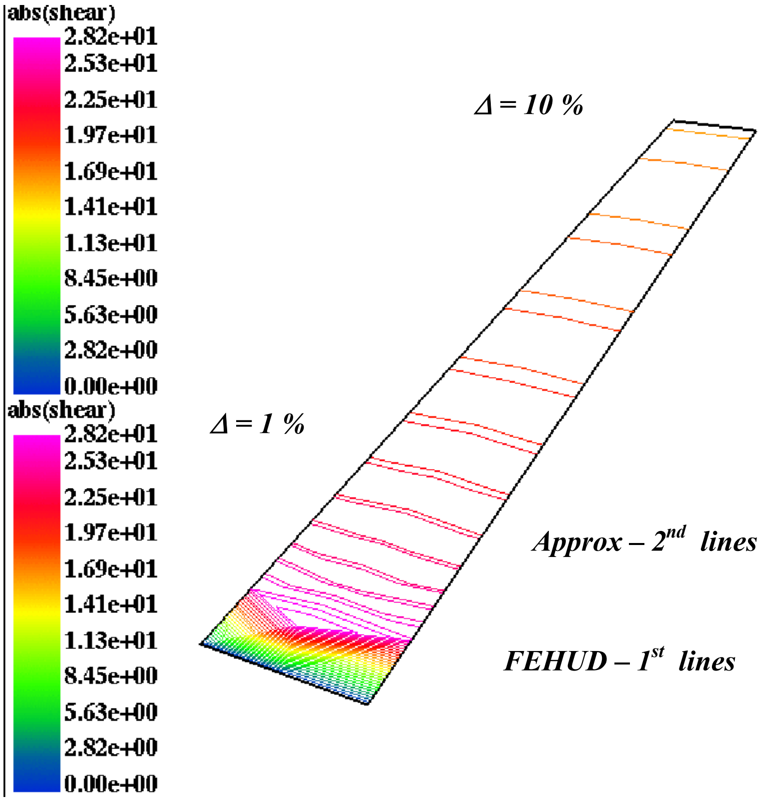


Fig. 3.1-3 Comparison of Abs – Shear contours

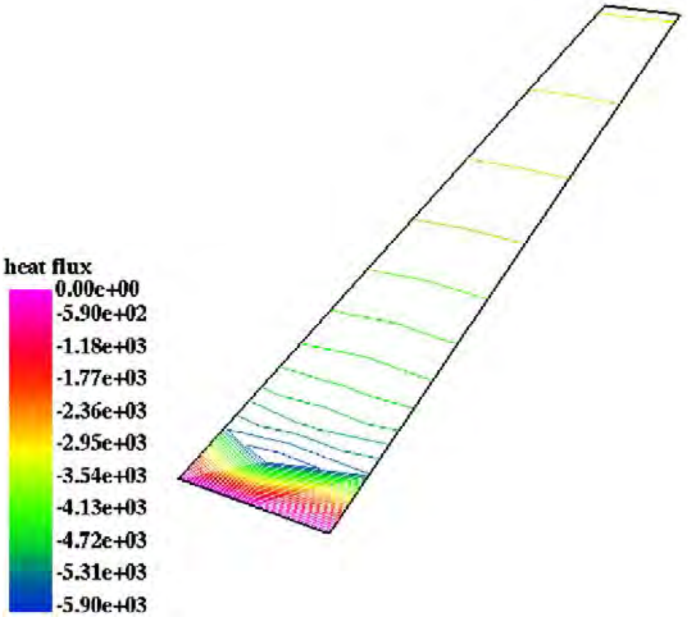


Fig. 3.1-4 FEHUD Heat Flux Contours

Configuration #1

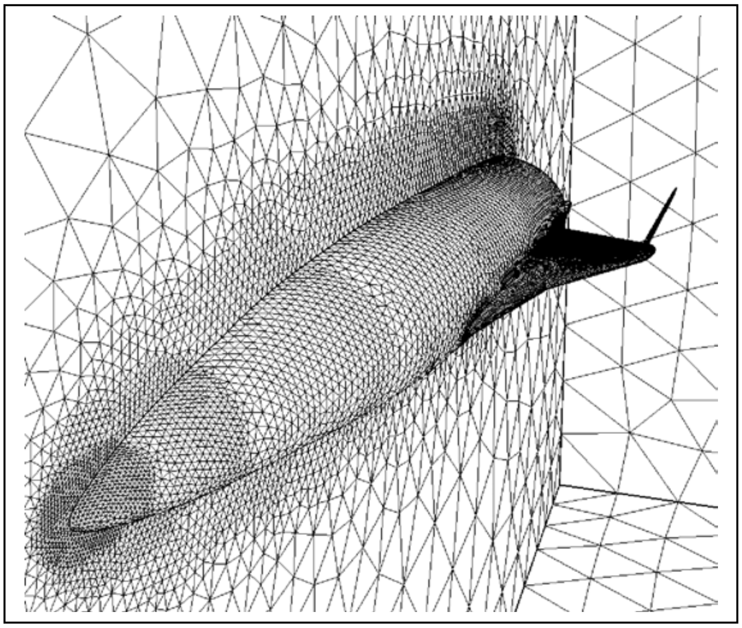


Fig. 3.1-5 CFD Mesh for Generic HRRLS Vehicle

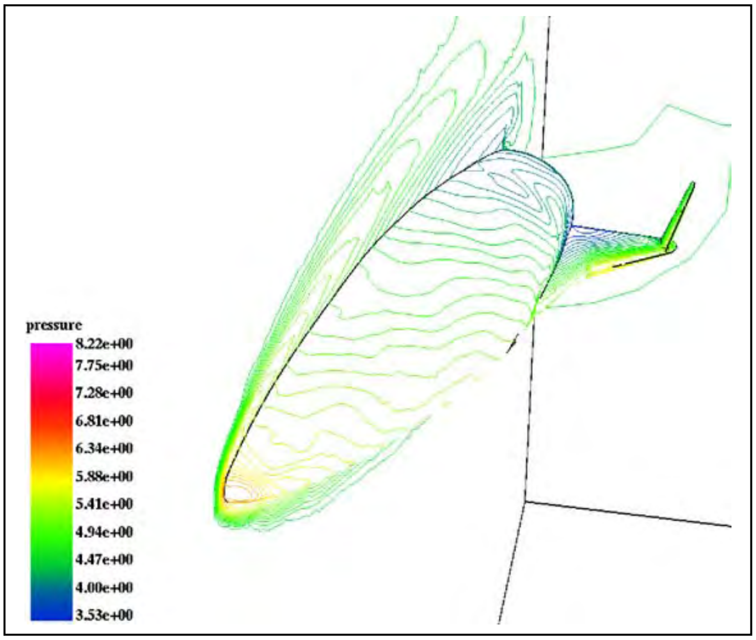


Fig. 3.1-6 Pressure Contours for Generic HRRLS Vehicle

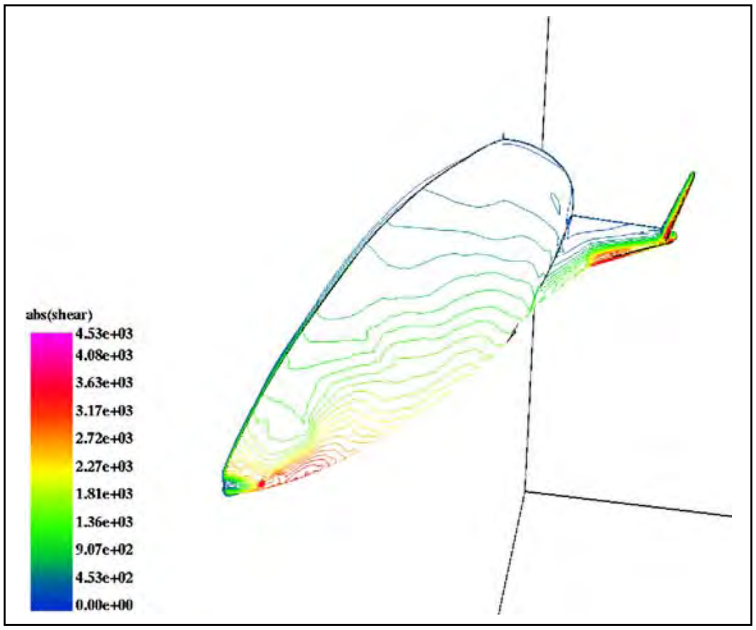


Fig. 3.1-7 Abs- Shear Contours on Generic HRRLS Vehicle

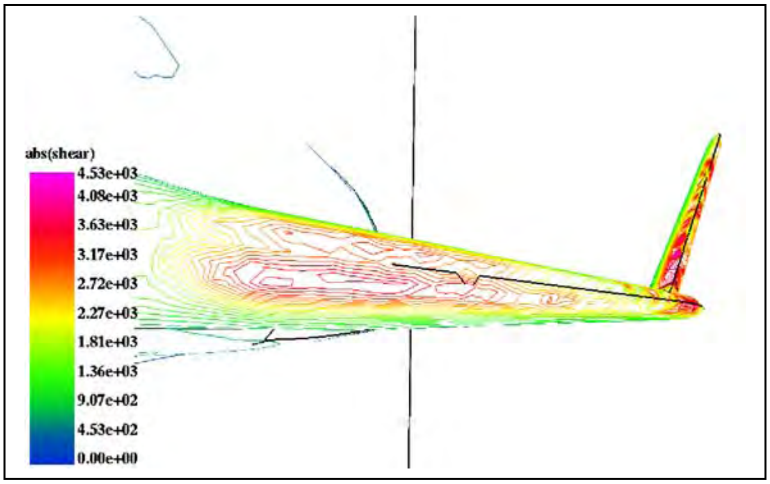


Fig. 3.1-8 Close Up of Abs –Shear on HRRLS Vehicle Wing

3.2 Euler + BL to Structures Integration

Configuration and Meshes

The configuration for this set of cases is shown in Fig. 3.2-1. Note that we used a different geometry than the previous case. This is identified in this report as "configuration #2".

Figure 3.2-2 shows the surface CFD mesh and Fig. 3.2-3 gives the symmetry plane CFD mesh and the surface FEAP structures mesh. Figure 3.2-4 shows the internal structures mesh which we created using beams and plates.

Configuration #2

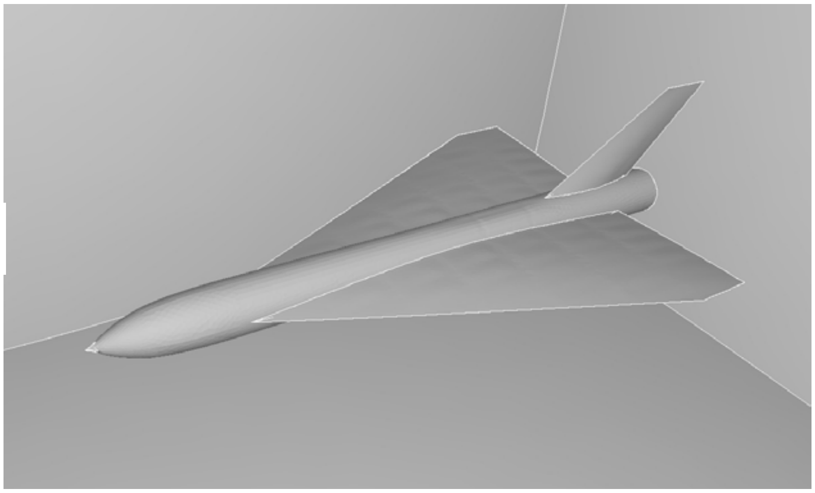


Fig. 3.2-1 Generic Vehicle Configuration #2

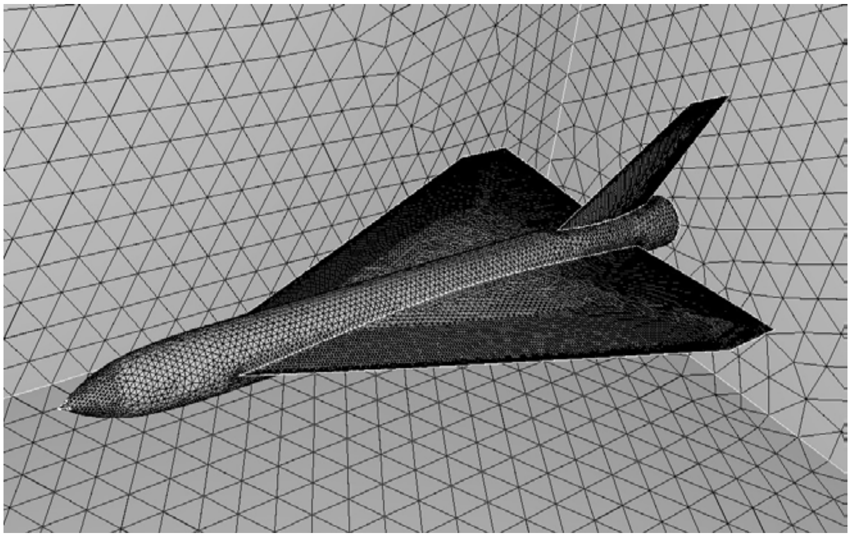


Fig. 3.2-2 CFD Surface Mesh

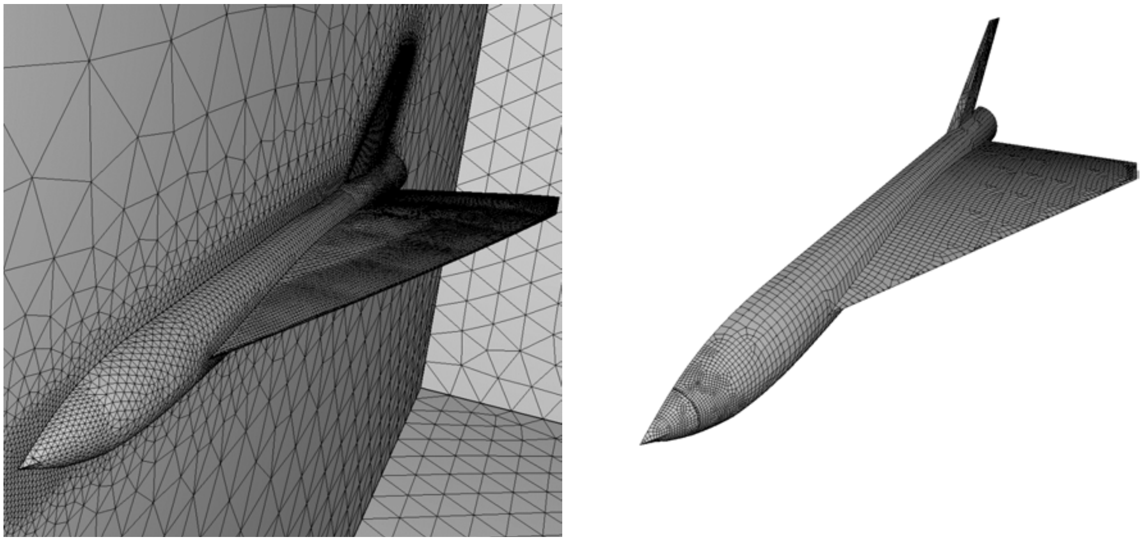


Fig. 3.2-3 CFD Symmetry Plane Mesh and External Structures Mesh

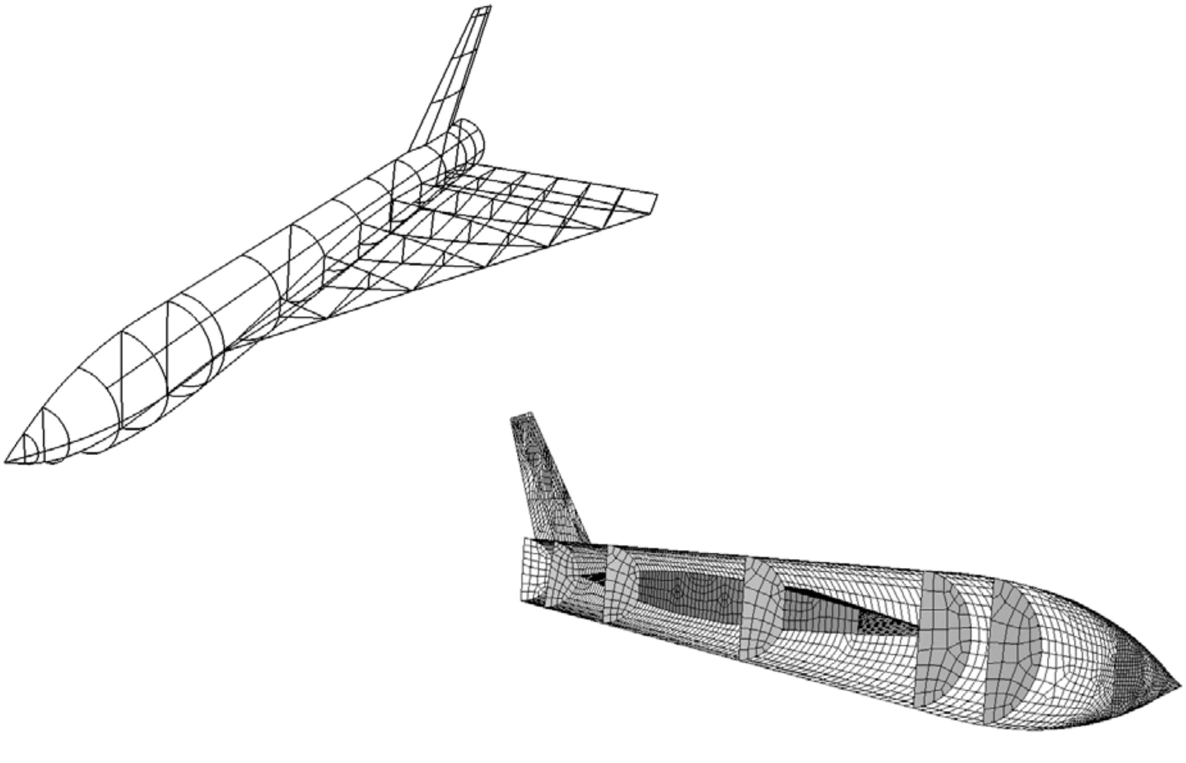


Fig. 3.2-4 Internal Structures Mesh

Solution - Coupling of Pressure and Shear Loads

This Section shows results of a run where we couple both the pressure loads and the shear loads to the structure. This run is also for the steady state case.

The case was run for 10 FEFLO – to –FEAP iterations. Figure 3.2-5 gives the convergence history. The maximum displacement change drops dramatically by 4 iterations and has completely stopped changing by the 10 iterations.

Figure 3.2-6 shows the original vehicle position (wireframe) and the final position on top (shaded). Figure 3.2-7 shows the vehicle displacement in enlarged views.

Figure 3.2-8 plots the final pressure contours and shear vectors at steady state.

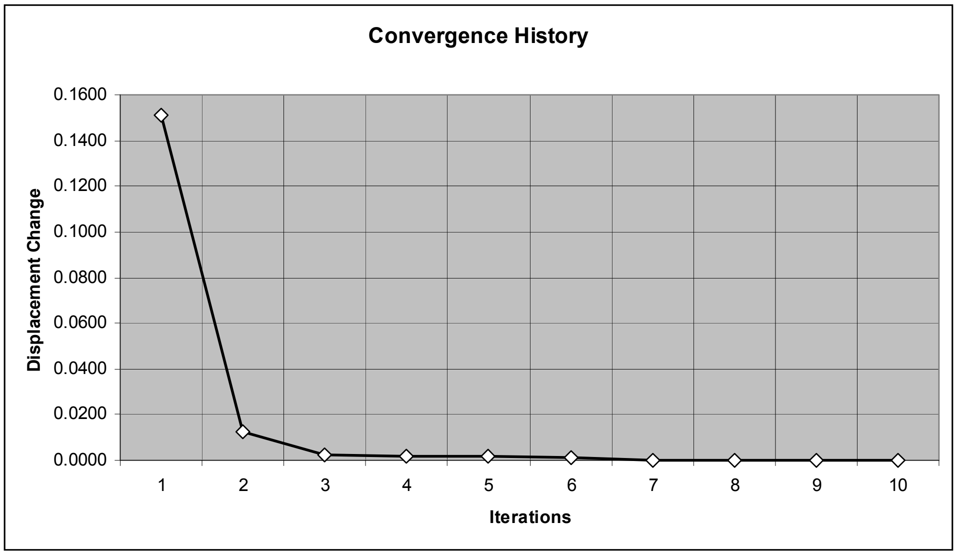


Fig. 3.2-5 Convergence History – Max Displacement Change vs. Iterations

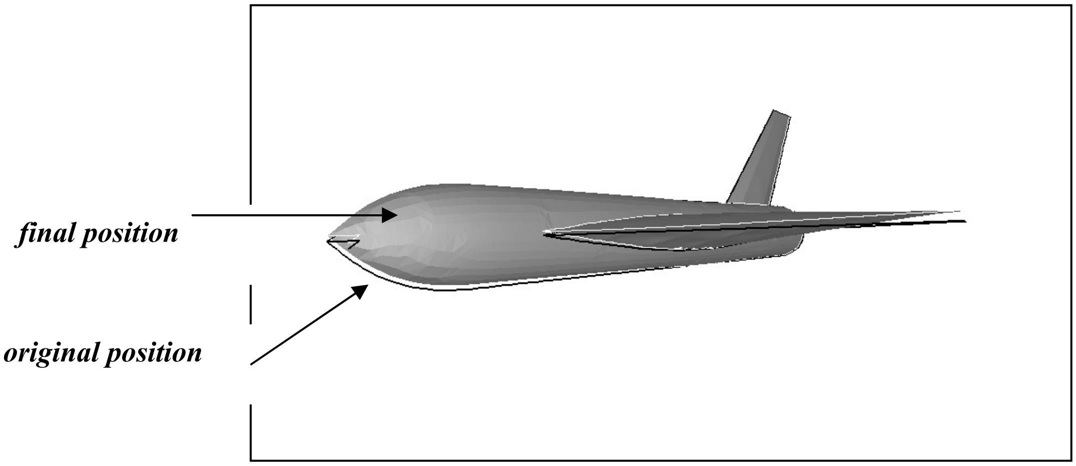


Fig. 3.2-6 Vehicle Displacement from Both Pressure and Shear Loadings

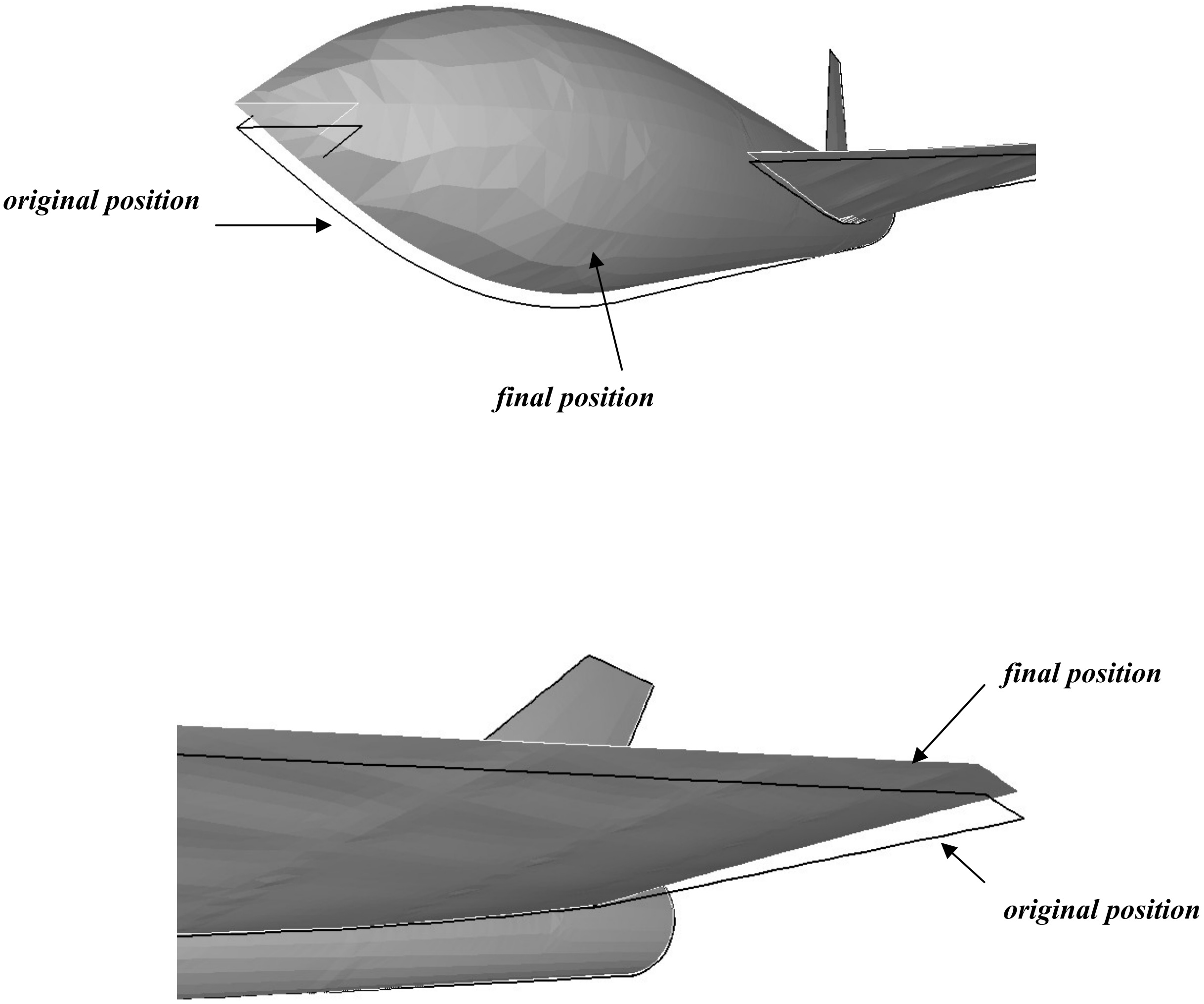
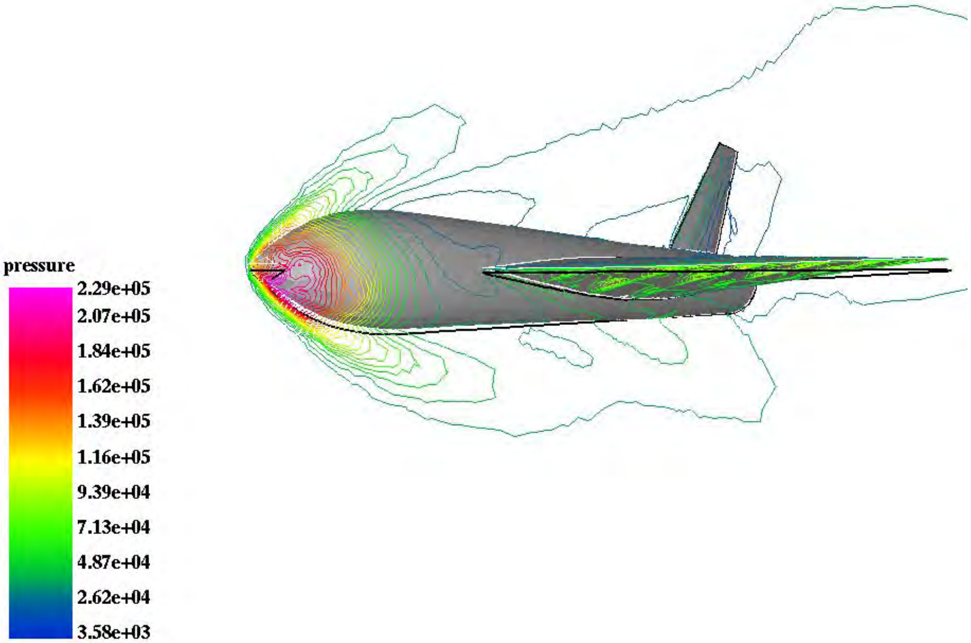
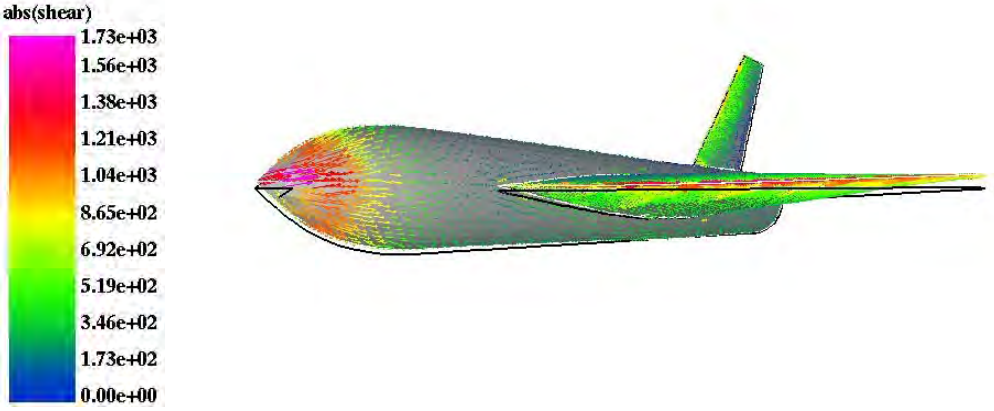


Fig. 3.2-7 Enlarged Views of Displacement
Line = original position, shaded = final position



Final Pressure Contours



Final Shear Vectors

Fig. 3.2-8 Final Pressure Contours and Shear Vectors at Final Position

3.3 Thermal Code Integration

Formulation

Accurate prediction of aerothermal heating is a key requirement when trying to design vehicles that operate in the hypersonic regime. The last three decades have witnessed the emergence of reliable, fast and easy-to-use computational thermo-dynamics (CTD) and computational fluid dynamics (CFD) codes. It is a part of our plan to couple these two disciplines. We choose the loose coupling techniques, whereby a flow solution, which yields heat fluxes, is followed by a heat transfer solution, which yields surface temperatures, and the cycle is repeated until convergence is achieved.

Let us consider the case of high Mach-number flow shown in Fig. 3.3-0. For the sake of simplicity, we can consider laminar flow. The loose coupling methodology works the same for turbulent flow as long as the heat flux is calculated properly. The local friction coefficient c_f and heat flux q across the wall for a flat plate at position L from the leading edge are given by [4]:

$$q = \alpha (T_w - T_s) = Nu \frac{\lambda}{L}(T_w - T_s)$$

$$Nu = 0.5 c_f \, \mathrm{Pr}^{1/3} \, \mathrm{Re}_L \ , \quad c_f = 0.664 \, (\mathrm{Re}_L)^{-0.5}$$

$$\mathrm{Re}_L = \frac{\rho V_\infty L}{\mu} \ , \ \frac{T_w}{T_\infty} = 1 + r\frac{(\gamma - 1)}{2} Ma^2 \ , \ r = 0.85 \ \mathit{to} \ 0.88 \tag{3.1}$$

Here α, Nu, λ, T_∞, T_w, T_s, Pr, Re_l, and r denote the film coefficient, Nusselt-number, conductivity of the fluid, free-stream temperature of the fluid, temperature of the fluid outside the boundary layer due to compression, temperature of the solid, Prandtl number, Reynolds-number and recovery factor respectively.

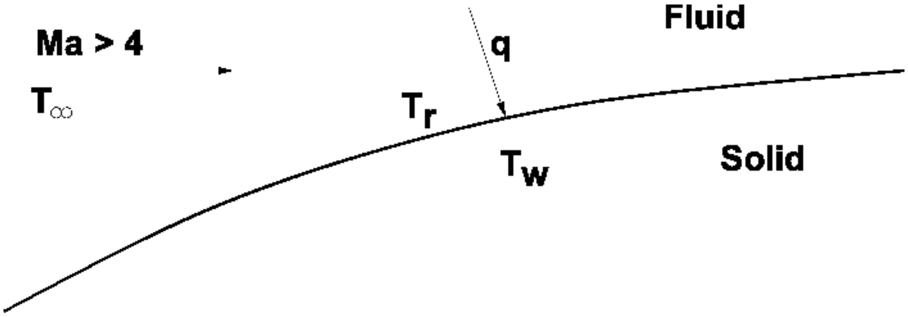


Fig. 3.3-0 Schematic of Euler + BL Flow with Heat Transfer

Let us consider now the common case of a thin shell of thickness δ, conductivity k, density ρ_s, heat capacity c_s and temperature T. Applying the heat load given by Eq.(3.1), we obtain, for a single time step:

$$\rho_s\, \delta\, c_s\, \frac{\Delta T}{\Delta t} = q = \alpha\,(T_w - T)$$

(3.2)

The assumption is that we have a thin shell, and that any heat transfer into the solid is small. Equation (3.2) is representative of what happens in loose coupling: the heat load is transferred to the solid, and a new temperature field is computed. Suppose, without loss of generality, that the solid wall is cold at the beginning. This will imply a large positive value for q. Depending on the time step chosen, the (correct) value of $T = T_w$ may not be reached due to instability. If the value of Δt is too large, we get a temperature $T > T_w$, which can lead to an even larger negative heat flux in the next time step. The condition for stability is given by:

$$\Delta t < \frac{2\rho_s\, \delta\, c_s}{\alpha}$$

(3.3)

which for a typical hypersonic case can be very restrictive. A better way to proceed is to transfer to the CTD code not only the heat flux of the fluid at the solid surface, but also the fluid temperature outside the boundary layer and the film coefficient. In this way, one can integrate implicitly:

$$[\frac{\rho_s\, \delta\, c_s}{\Delta t} + \alpha\,] = q = \alpha\,(T_w - T)$$

(3.4)

This removes the time step size restriction given by Eq. (3.3). This option was implemented in FEMAP, FEFLO and FEHEAT.

Solutions

Figure 3.3-1 shows a hypersonic flyer configuration identified in this report as "configuration #1". This case is for Mach = 10. The length is approximately 20 m, and the incoming flow conditions were set to: (all units kg/m/sec).

$$\rho = 0.4135 \ , \quad v = (2986.0\ ,261.0\ ,0.0), \quad p = 26583.0\ , \quad \mu = 1.73e-5\ , \quad Pr = 0.71\ , \quad cp{=}1000$$

The shell thickness was δ = 0.01 and material parameters typical of Aluminum were chosen.

$$\rho_s = 2700.0 \ , \quad c_s = 0.91 \ , \quad k = 250.0.$$

The shear and heat transfer were computed using the approximate formulas for turbulent flow given by Oktay [3] and Schlichting [4].

Both FEFLO and FEHEAT were run to steady state in each coupling step. For FEHEAT, a time step of $\Delta t = 1$ sec was used, 20 times larger than the explicit time stepping limit given by Eq. (3.3).

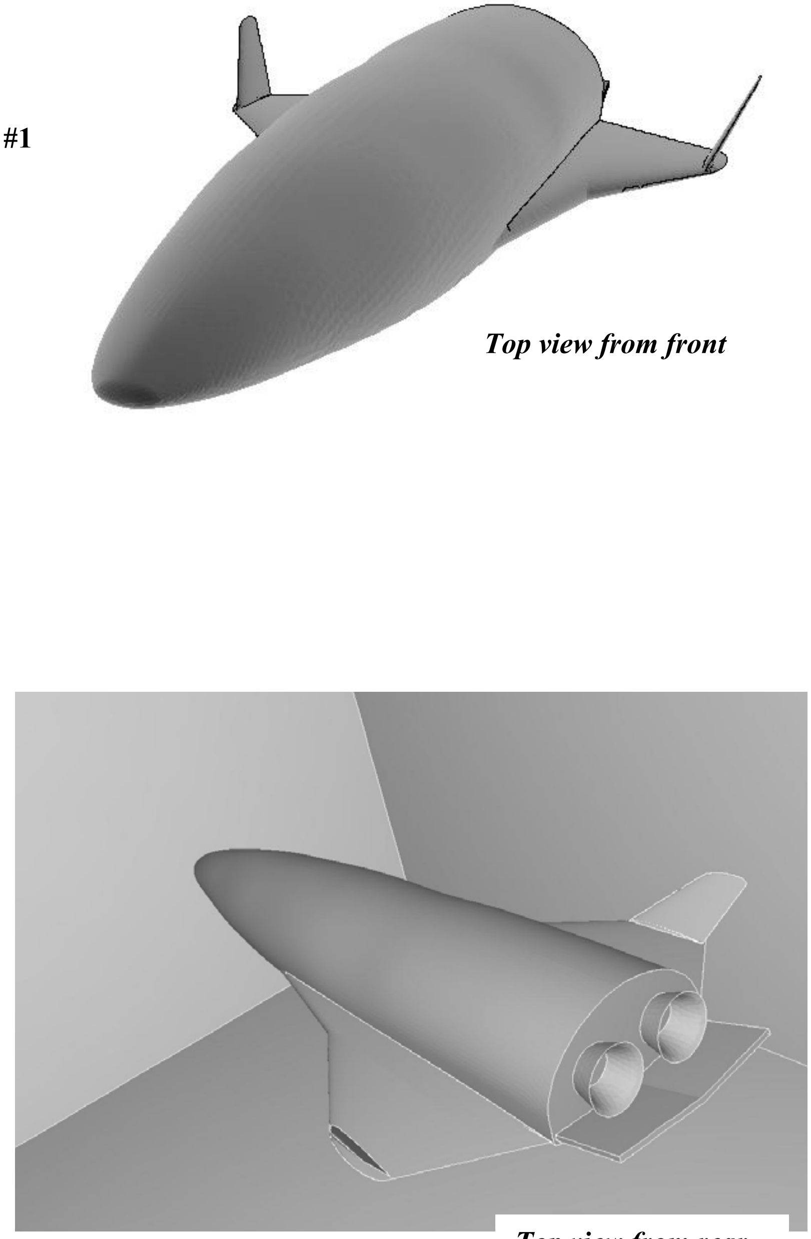
In order to estimate the explicit timestep required for loose coupling, consider that: $\alpha = O(10^3)$, $\rho_s = 2.7 \cdot 10^3$, $\delta = 0.01$, $c_s = 0.91$. This gives $\Delta t < 0.054$ sec.

The CFD and heat transfer surface meshes are shown in Fig. 3.3-2. The Euler solution for Pressure and Mach number from a top and bottom view is shown in Figs. 3.3-3 and 3.3-4.

Figure 3.3-5 shows the points which were selected for computing streamlines to pass to the boundary layer code FEHUD. The computed streamlines are shown in Fig. 3.3-6.

The converged solution is shown in Figs. 3.3-7 through 3.3-10 for four different views. The surface density, velocity magnitude, heat flux magnitude, shear magnitude, and solid surface temperature are shown.

Configuration #1



Top view from front

Top view from rear

Fig. 3.3-1 Hypersonic Flyer Configuration #1

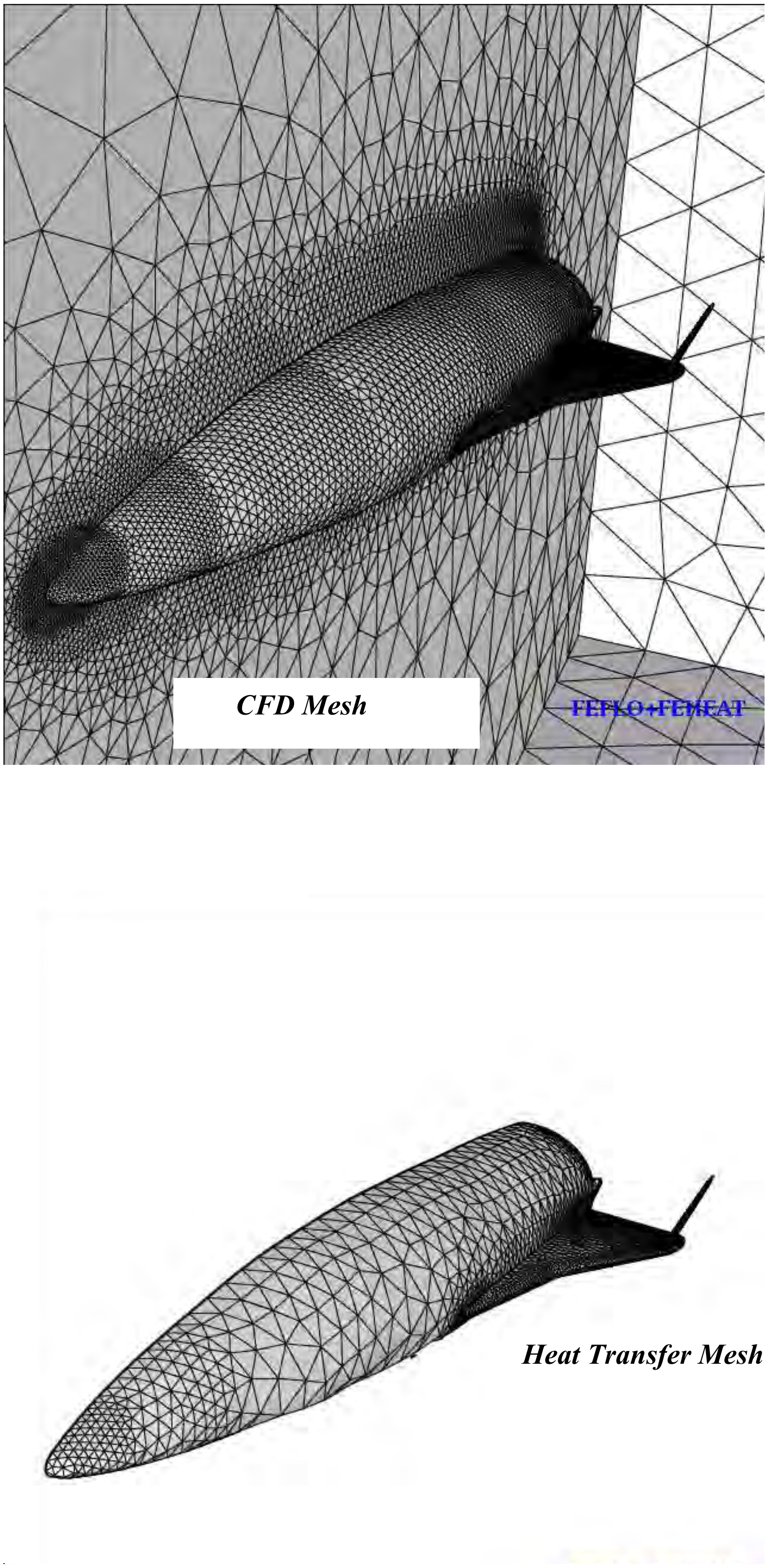


Fig. 3.3-2 CFD and Heat Transfer Meshes

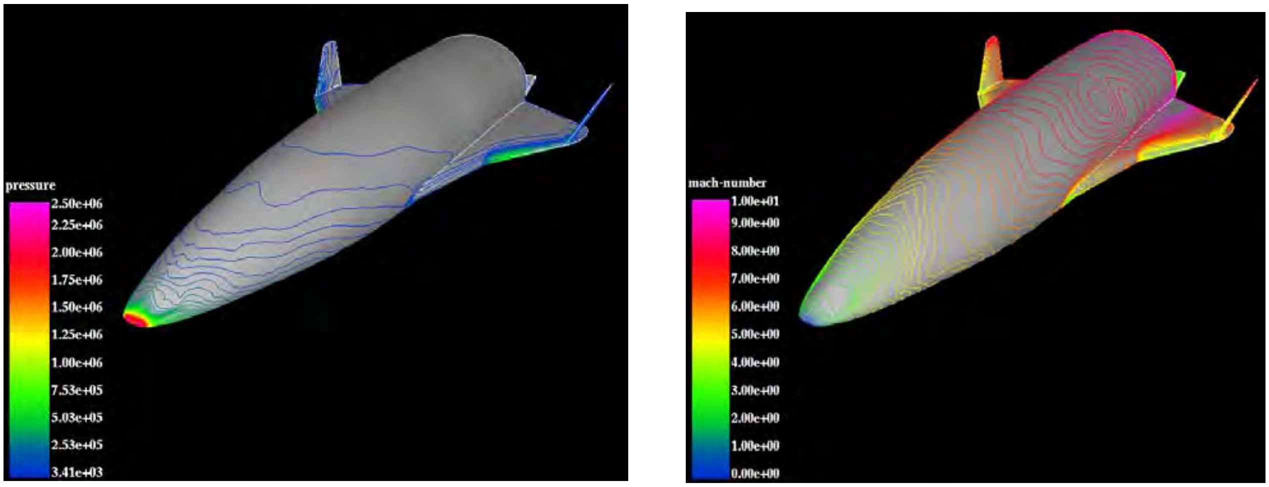


Fig. 3.3-3 Euler Solution – Pressure and Mach - Top

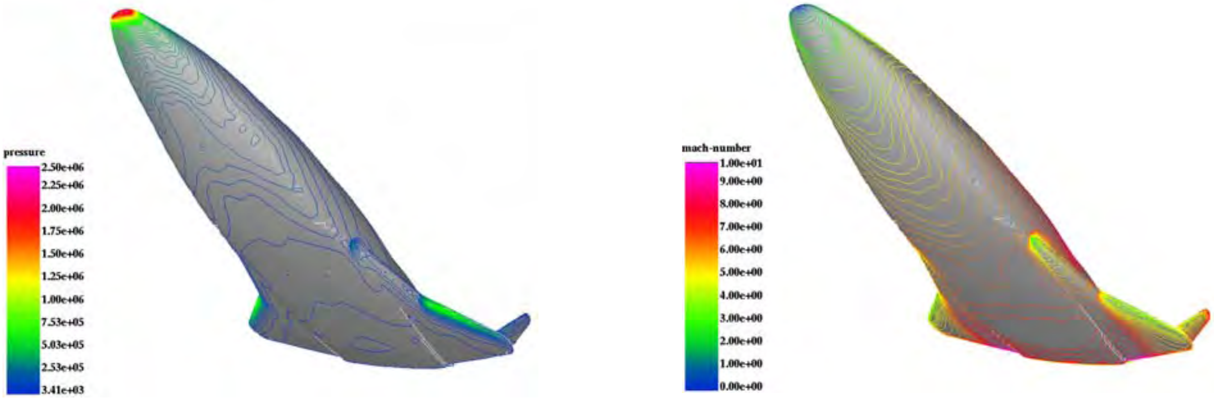


Fig. 3.3-4 Euler Solution – Pressure and Mach - Underneath

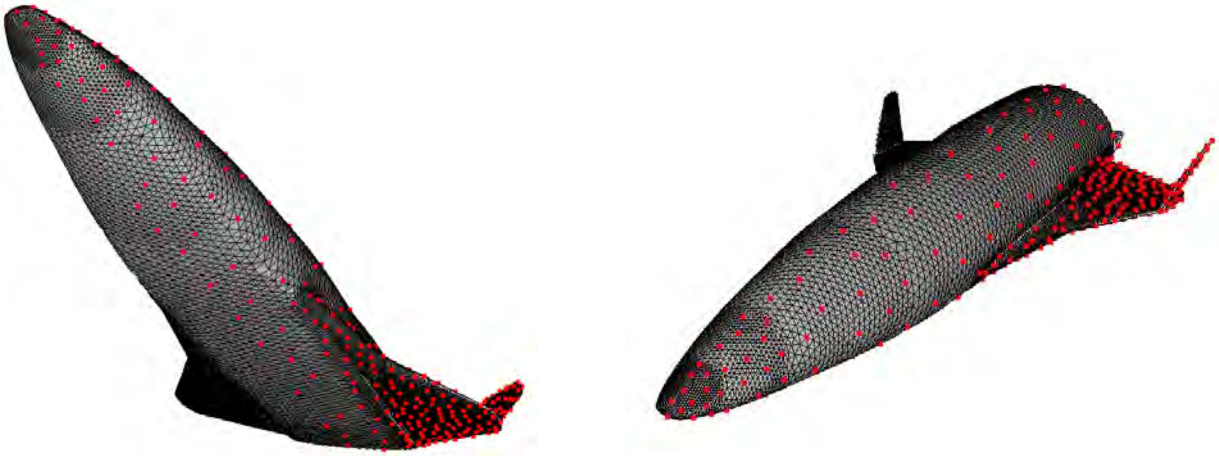


Fig. 3.3-5 Points used for Generating Streamlines

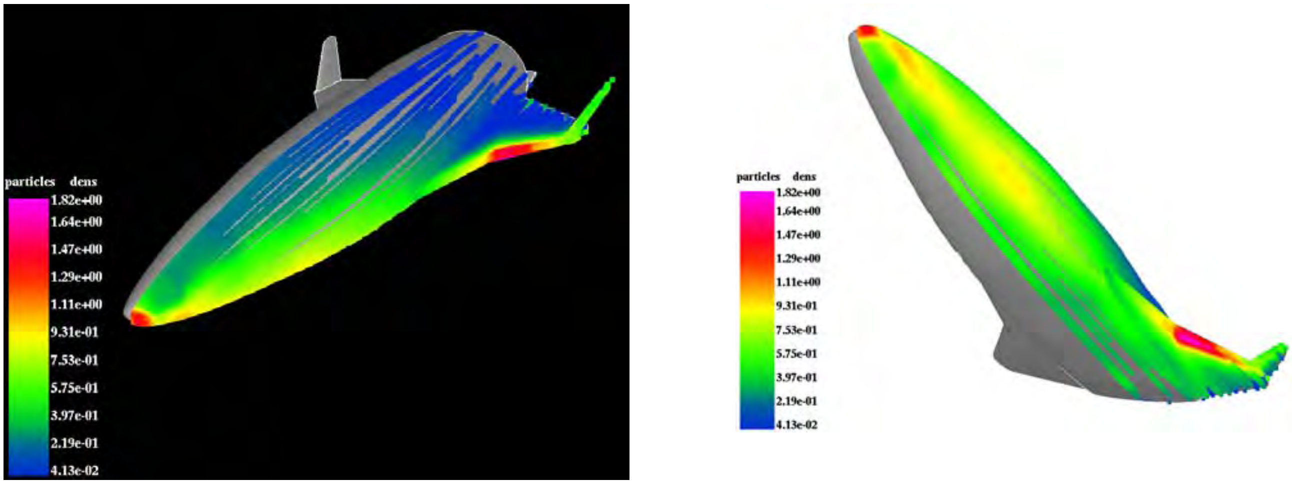


Fig. 3.3-6 Streamlines used in HUD BL Code

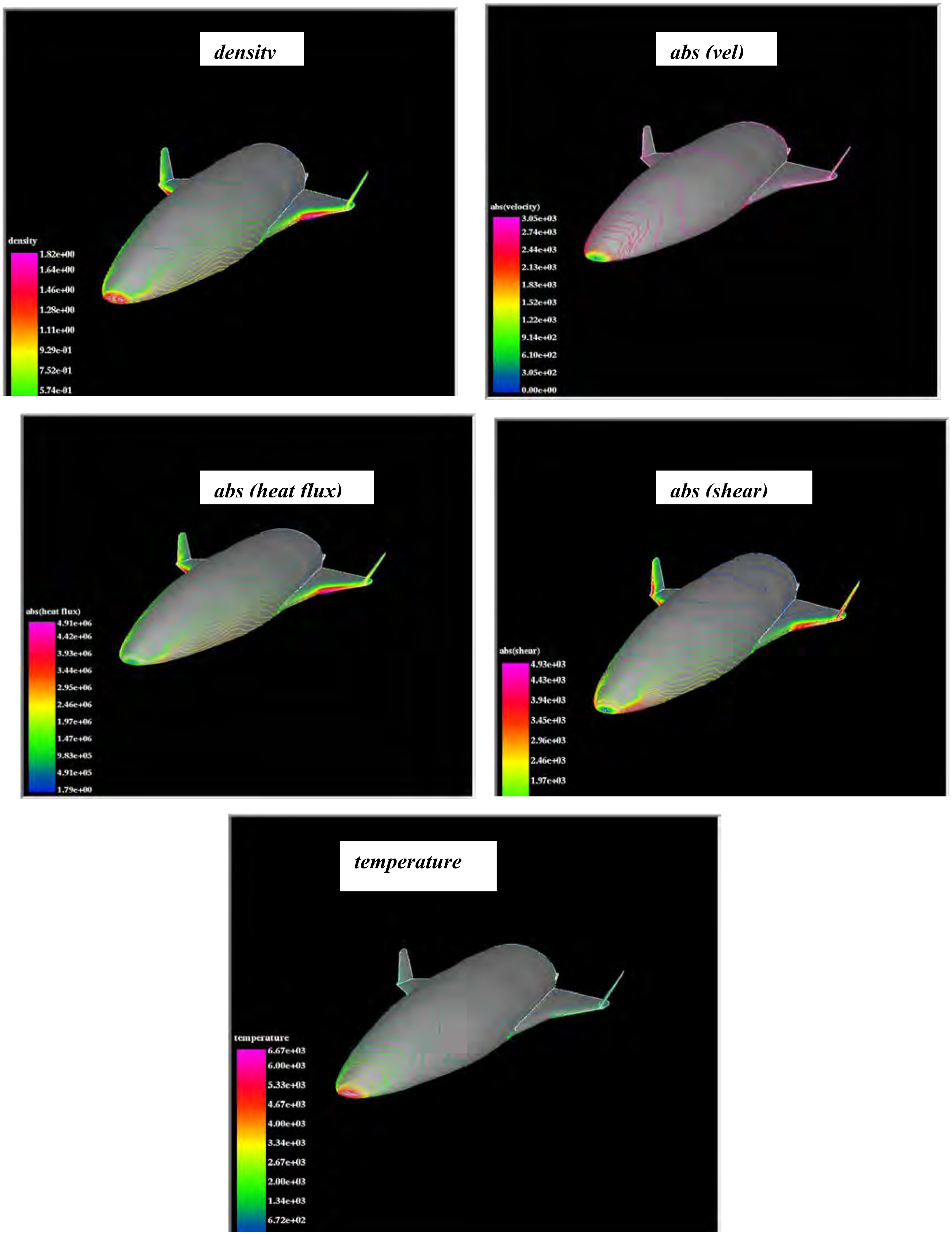


Fig. 3.3-7 Top View from Front

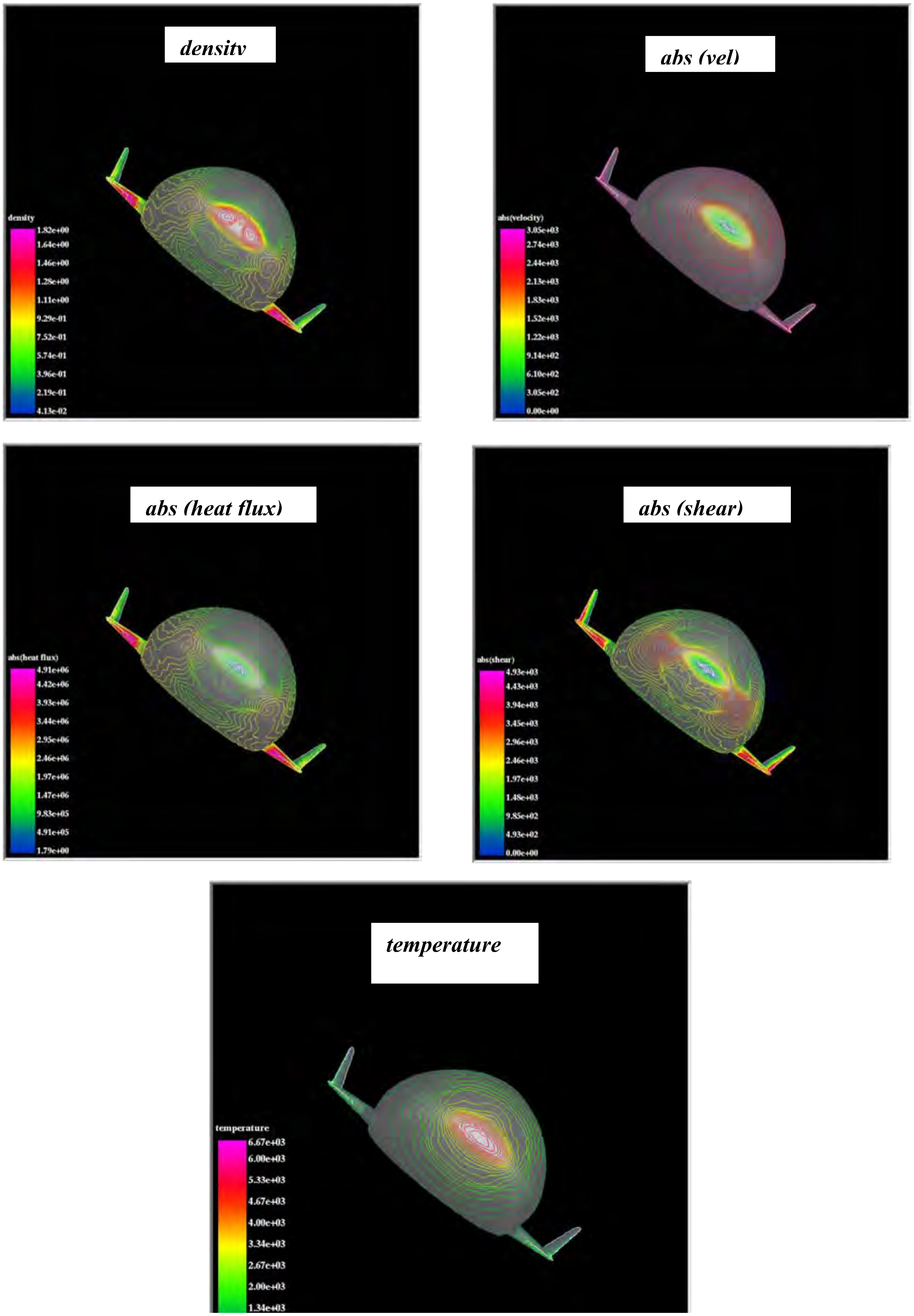


Fig. 3.3-8 Front View

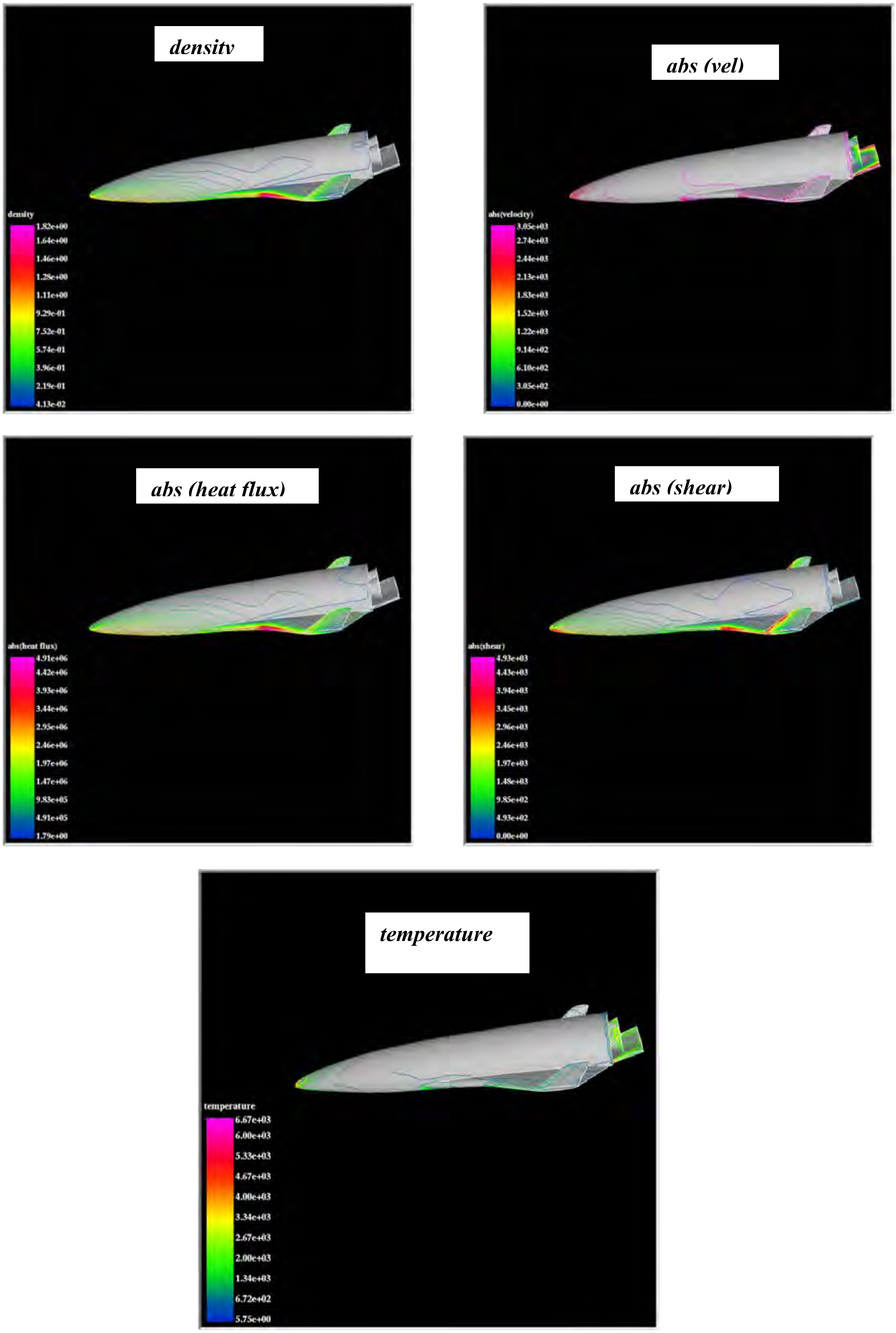


Fig. 3.3-9 Side View

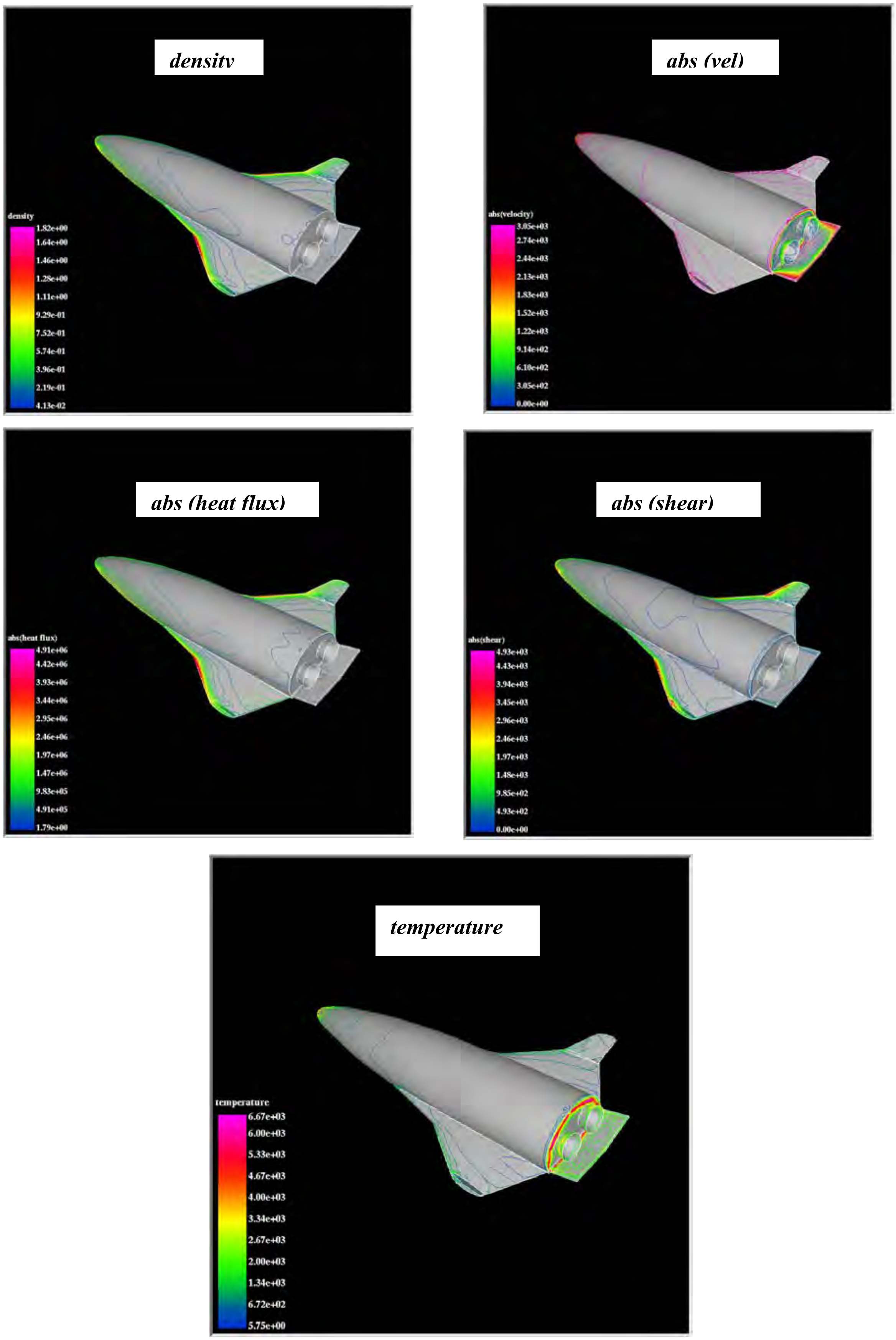


Fig. 3.3-10 Top View from Rear

3.4 References

1. Pinckney, S.Z., “Method for Predicting Compressible Turbulent Boundary Layers in Adverse Pressure Gradients”, NASA TM X-2302, August 1971.

2. Spradley, Lawrence, Rainald Lohner, and Pinckney, S. Z., “Integration of Three-Dimensional CFD with Combustion Cycle Analysis”, JANNAF Paper APS-VI-06, 35th Combustion / Airbreathing Joint Meeting, Cocoa Beach, FL, October 1999.

3. Oktay, E. Asma, C. O. Fast Drag Prediction Method Using Euler Equations, Journal of Spacecraft and Rockets Vol. 37, No. 5, September-October 2000.

4. H. Schlichting - Boundary Layer Theory; McGraw-Hill (1979).

4.0 TETRAHEDRAL MESH ALGORITHM IMPROVEMENTS

4.1 Background

Finite volume CFD methods have proven to be accurate and reliable for computing heat transfer in turbulent flow fields for subsonic through supersonic flow regimes. These methods generally apply a one-dimensional solution to an approximate Riemann problem to evaluate the required flux functions for the finite volume formulation. However, recent studies have revealed serious deficiencies in these schemes for flows at hypersonic speeds.

One of the shortcomings is that the methods place conflicting demands on the numerical flux functions in the finite volume method. For hypersonics, the schemes must have sufficient dissipation to capture the strong bow shocks on the forebody without developing overshoots and oscillations in the vicinity of the shock. However, the scheme must also have minimal dissipation in the boundary layer such that the effects of the physical viscosity are not swamped by numerical dissipation which causes serious errors in the surface heating evaluations. This last requirement is equivalent to requiring that the dissipation in the linear fields must vanish as the velocity normal to the wall vanishes. The simultaneous satisfaction of these two requirements makes computing hypersonic turbulent flows very challenging for both the forebody flow field and the base region of the vehicle.

For unstructured meshes which are not aligned with the bow shock, cross coupling between the Riemann problems in the different directions occurs. The strong jump in velocity in the shock direction is interpreted by the current Riemann solvers, in the non-aligned directions, as strong jumps in the other characteristic variables. This causes trouble for schemes which have numerical dissipation which scales directly with the velocity normal to the cell interfaces in directions tangential to the shock. The lack of dissipation in this direction means that the perturbations due to the cross coupling are either 1) amplified for schemes based on the use of anti-diffusion or, 2) are neutrally stable which means that the disturbances propagate without damping.

In both cases, the solution quality is severely degraded and in many cases actually wrong. Ad hoc fixes to the eigenvalues have been used by some investigators, but all such approaches are problematic in satisfying the conflicting requirements for hypersonic shock capturing. In addition, these entropy corrections merely attempt to add dissipation rather than dealing with the anti-diffusion terms directly. This effect can be especially aggravated in the case of unstructured grids where the cell interface will usually be highly misaligned with the shock direction.

We presented solutions at the Air Force / NASA Foundational Hypersonics Review in Virginia Beach on a Mach 17 blunt body case [1]. The post-shock and boundary layer regions were analyzed by viewing isosurfaces of velocity magnitude in a region just behind the shock and another in the high shear region of the boundary layer. The plots in Fig. 4.0 show the correct flow in the regions outside of the boundary layer. However, the boundary layer flow itself is erroneous. Several Riemann solvers were tried with all producing similar results. This points directly at the flux function evaluation in regions of high shear such as hypersonic boundary layers.

All flow variables were inspected and displayed in a cylindrical surface that is just outside the boundary layer region. Surprisingly, this showed that the results outside the boundary layer region do not exhibit the irregularity observed in the shear stresses and heat fluxes on the body surface. Figure 4.0 is an iso-surface plot of the velocity magnitude. The left side is outside of the boundary layer and the right side is inside of the boundary layer region, not on the wall itself. Note the "shift" in the contours, especially near the top and bottom of the iso-surface. This indicates erroneous numerical evaluation of some type. Therefore, we believe that the region of erroneous numerics must be located in the boundary layer region and not in the isotropic mesh region.

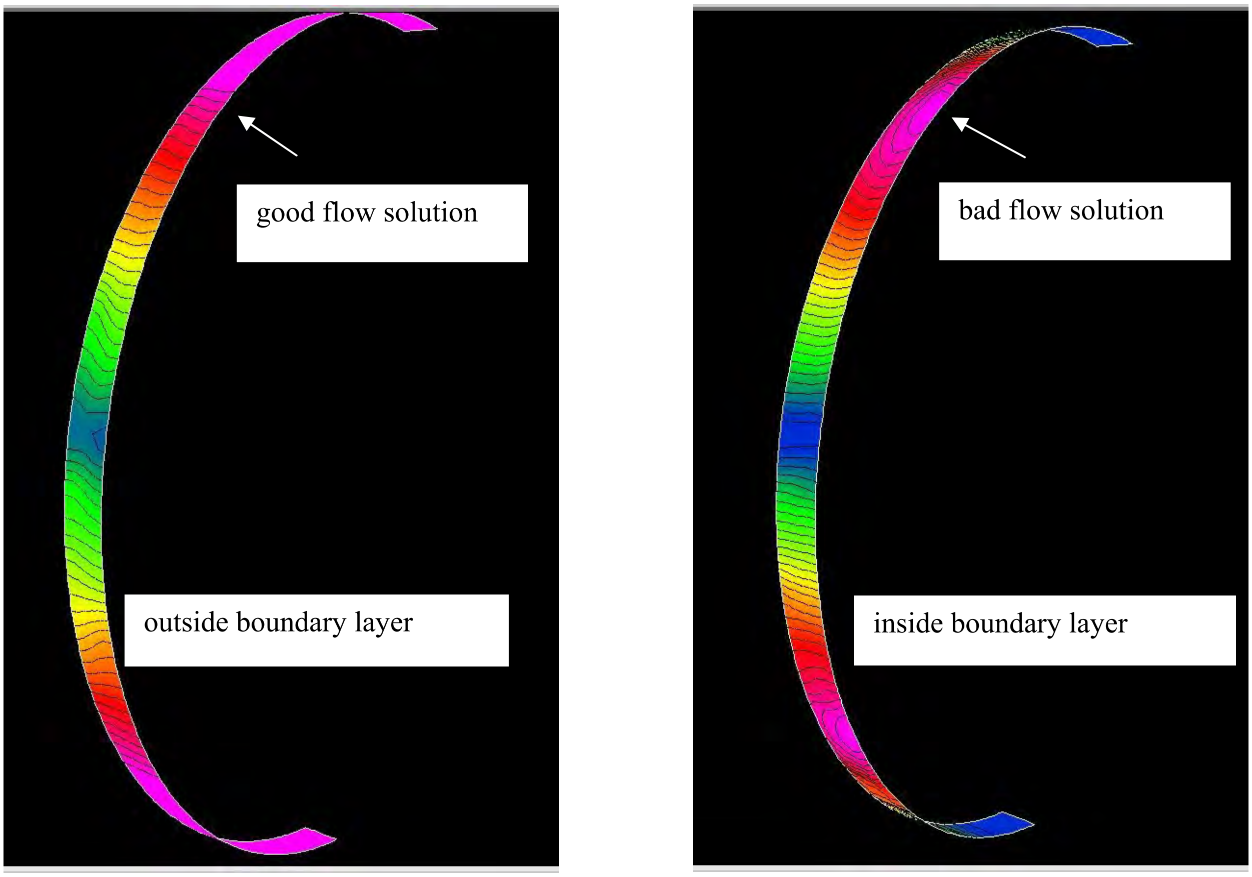


Fig. 4.0 Iso-surfaces of Velocity Magnitude for Mach 17 Blunt Body Case

Gnoffo [2] presented results showing the same type of behavior which we have observed. His tetrahedral mesh solutions clearly point to the evaluation of the flux functions at cell faces as a major difficulty. In the paper, he attempts to reconstruct the flux functions by solving the full three-dimensional Riemann problem at the cell faces. This would eliminate the one-dimensional assumption as the main difficulty. However, the evaluation of the full three-dimensional fluxes at all cell faces proved to be very computationally expensive. Results showed improvement in spanwise uniformity of heating, but still had accuracy issues near the point of peak heating with as large as 25% discrepancy compared to a benchmark.

Candler [3] presented results, at this same conference in Virginia Beach, of his work on missile base flows in hypersonics. His work differs from Gnoffo [2] and our work [1] in that he uses structured meshes entirely. The work is teamed with Holden [4] who has an extensive data base from the CUBRC facility. Computational results from the Candler code were compared to the measured data without good agreement in the base area. Specifically, he compared solutions using finite volume methods, with one-dimensional approximate Riemann solvers, to a classic centered scheme with damping. The centered scheme results actually agreed with the data better than the directional-dependent finite volume solutions. He concluded that the current finite volume methods with the various Riemann solvers are not suitable for accurate base flow computation in hypersonics. It is important to note here that these solutions were for structured meshes.

Gnoffo [5] recently presented updates to his previous flux reconstruction method. The idea is to split the flux functions into three principal directions, thus better aligning the flow and the mesh. He also replaced the original formulation for the primitive variable gradients at the nodes with a new formulation. The net computing penalty is about 3 as compared to the one – dimensional flux method. There are apparently some additional problems with the flux limiter algorithms used, but his results are very encouraging.

In the remainder of this Section, we present our FEFLO implementation of the Gnoffo method [5]. We then give results of our computation of the Mach 17 blunt body case. We compare solutions for our original HLLC algorithm with solutions from our new implementation of the Gnoffo algorithm.

4.2 FEFLO Implementation of Gnoffo Method

The main emphasis has been the implementation of Gnoffo's Multi-Dimensional Flux Reconstruction Scheme for Hypersonic Simulations [5] on Tetrahedral Grids within FEFLO [1].

Consider the system of conservation laws given by:

$$\mathbf{u}_{,t} + \nabla \cdot \mathbf{F} = 0 \tag{4.1}$$

Here **u** denotes the vector of conserved variables and **F** the flux tensor. Denoting by N^i the shape functions associated with a node, a Galerkin discretization in space yields:

$$\int_{\Omega} N^i N^j \hat{\mathbf{u}}_{j},t d\Omega + \int_{\Omega} N^i \nabla \cdot N^j \hat{\mathbf{F}}_j d\Omega = 0 \tag{4.2}$$

All integrals are evaluated using the element subdomain paradigm

$$\int_{\Omega} \cdot\cdot = \sum_{el} \int_{\Omega_{el}} \cdot\cdot \tag{4.3}$$

For finite element codes based on linear elements (such as FEFLO), it is convenient to switch from element-based loops to edge-based loops. Both the operation count and the number of indirect address fetches and stores are reduced significantly [1]. One can also show that this is equivalent to a so-called median-dual finite volume scheme.

Using the notation of Fig. 4.1, the conventional edge-based solver for the edge 1-3 would use a flux of the form:

$$\mathbf{f}_{AcB} = \frac{1}{2}\left[\mathbf{f}_3 + \mathbf{f}_1 - \mathbf{R}_c|\Lambda_c|\mathbf{R}_c^{-1}(\mathbf{q}_3 - \mathbf{q}_1)\right] \tag{4.4}$$

As stated by Gnoffo [2, 5] "The basic idea in multi-dimensional reconstruction is that: (1) reconstruction directions should be based on local flow characteristics and not be constrained by the grid; and (2) the flux components for any face will utilize a stencil employing all nodes of a surrounding element. Conventional, node-based schemes loop over edges when using quasi-one-dimensional reconstruction. In contrast, the new multi-dimensional reconstruction loops over elements."

Consider the two-dimensional element defined by nodes (1, 2, and 3) with centroid A in Fig. 4.1.

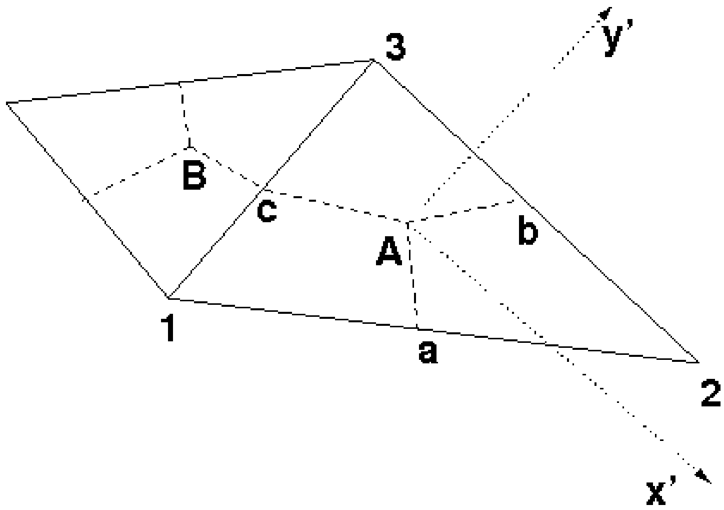


Fig. 4.1 Median Dual Finite Volume Scheme

A principal direction in the element, x', is defined by the direction of $\nabla\rho$. If density is locally constant, then the principal direction aligns with the local velocity computed as the average of nodal values. The inviscid flux within the element is then computed along and orthogonal to the principal flow direction.

Denoting by N^i the shape-functions within an element, we have:

a) Directions:

$$\mathrm{s}_1 = \frac{\nabla\rho}{|\nabla\rho|} \quad ; \quad \nabla\rho = \nabla N^i \hat{\rho}_i$$

$$\mathbf{t} \;:\; \mathrm{s}_1 \cdot \mathbf{t} > 0.1$$

$$\mathrm{s}_3 = \frac{\mathrm{s}_1 \times \mathbf{t}}{|\mathrm{s}_1 \times \mathbf{t}|}$$

$$\mathrm{s}_2 = \mathrm{s}_2 \times \mathrm{s}_1$$

(4.5-4.8)

b) Shape Function Derivatives in Directions 1-3:

$$N^i_{s_j} = \mathrm{s}_j \cdot \nabla N^i = \mathrm{s}^k_j N^i_{,k}$$

(4.9)

Therefore, we have:

$$\frac{\partial \mathbf{f}_{x',A}}{\partial x'} = N^i_{s_1}\mathbf{f}_i = \frac{1}{\Delta x'}\left[\beta^i_{s_1}\mathbf{f}_i\right]$$

(4.10)

The distance Δx' can be interpreted as a distance across the element in the x' direction, and is obtained from:

$$\frac{1}{\Delta x'} = \sum_i |N^i_{s_1}|$$

(4.11)

We then upwind (or 'stabilizes' the fluxes) using the so-called virtual node averaging, which proceeds as follows:

$$\mathbf{q}_{R,x'} = \sum_i \left(|\beta^i_{x'}| + \beta^i_{x'}\right)\mathbf{q}_i$$

$$\mathbf{q}_{L,x'} = \sum_i \left(|\beta^i_{x'}| - \beta^i_{x'}\right)\mathbf{q}_i$$

$$\mathbf{f}_{x'} = \frac{1}{2}\left[\mathbf{f}(\mathbf{q}_{L,x'}) + \mathbf{f}(\mathbf{q}_{R,x'}) - \mathbf{R}_{x'}|\Lambda_{x'}|\mathbf{R}^{-1}_{x'}(\mathbf{q}_{L,x'} - \mathbf{q}_{R,x'})\right]$$

(4.12-4.13)

The last expression stands for an approximate Riemann solver. Our Code FEFLO [1] allows for van Leer, Roe, HLLC, CUSP, AUSM+, and Godunov schemes.

If we consider again the original Galerkin RHS, we have:

$$\mathbf{r}^i = -\int_\Omega N^i \nabla \cdot N^j \hat{\mathbf{F}}_j d\Omega$$

(4.14)

Integration by parts yields

$$\mathbf{r}^i = \int_\Omega N^i_{,k}\hat{\mathbf{F}}^k d\Omega - int_\Gamma N^i \hat{\mathbf{F}}^k \cdot \mathbf{n}_k d\Gamma$$

(4.15)

The volume integral is integrated on the element level using the principal directions:

$$\int_\Omega N^i_{,k}\hat{\mathbf{F}}^k d\Omega = \int_\Omega N^i_{,k'}\hat{\mathbf{F}}^{k'} d\Omega = \int_\Omega (\mathbf{s}_{k'}\nabla N^i)\hat{\mathbf{F}}^{k'} d\Omega$$

(4.16)

4.3 Results and Comparisons

The blunt body at Mach 17 solution is compared for an HLLC versus Gnoffo flux evaluation method. The flow variables, pressure, Mach number, temperature and velocity vectors are compared first. The original HLLC flux method in our FEFLO code and the new GNOFFO flux algorithm are then compared for heat flux and shear stress. Figure 4.2 shows the surface of the mesh in full view, close up, and zoom views. The incoming flow conditions are:

- cylinder diameter: d = 1.0 m
- density : ρ = 0.100000E - 02 kg/m3
- velocity: v = 0.500000E + 04 m/sec
- pressure : p = 57.428571 N/m2
- viscosity: μ = 0.1850E - 04 Pa
- temperature: T = 200 K
- Sutherland temperature: Ts = 110 K
- Prandtl-nr: Pr = 0.71
- Specific heat: cp = 1005
- Temperature of the wall: Tw = 1000 K

The code solutions are now stable using both the first and second order schemes. We show the latest results for the Ma=17 viscous case here. The solution using the original HLLC method is shown on the left side and the Gnoffo flux solver on the right hand side for the basic flow variables. Pressure and Mach number are shown in Fig. 4.3 and temperature and velocity vectors for steady state in Fig 4.4. A zoomed view, *near* the body surface, of the velocity vectors and the density is shown in Fig. 4.5 for both HLLC and Gnoffo methods. Pressure and density *on* the body surface are shown in Fig. 4.6 for both methods.

Heat flux and absolute value of shear stress solutions from the 2nd order FEFLO HLLC flux solver are compared to our implementation of the Gnoffo flux solver in Figs. 4.7 and 4.8 respectively. Note that these plots are not to the same scale. We plotted each one from min to max to fully illustrate the features for comparison to each other. A primary interest is the jagged behavior.

We see little improvement in the smoothness of the heat flux, but the magnitude of the stagnation point heating was dramatically affected. The peak heating numbers at the stagnation point compare as follows:

Original HLLC method	q = ~29. E+5	J / (m2-sec).
Gnoffo flux solver	q = ~8.8 E+5	J / (m2-sec).

The Gnoffo method reduced the peak heating rate by about ~3 (shown in Fig. 4.7). The shear stress is greatly improved. The plot in Fig. 4.8 shows that it is now almost symmetric using the Gnoffo method. This is a significant improvement over the very jagged results obtained with the standard HLLC method. The magnitude of the shear is reduced by about ~ 2.5 with the Gnoffo method.

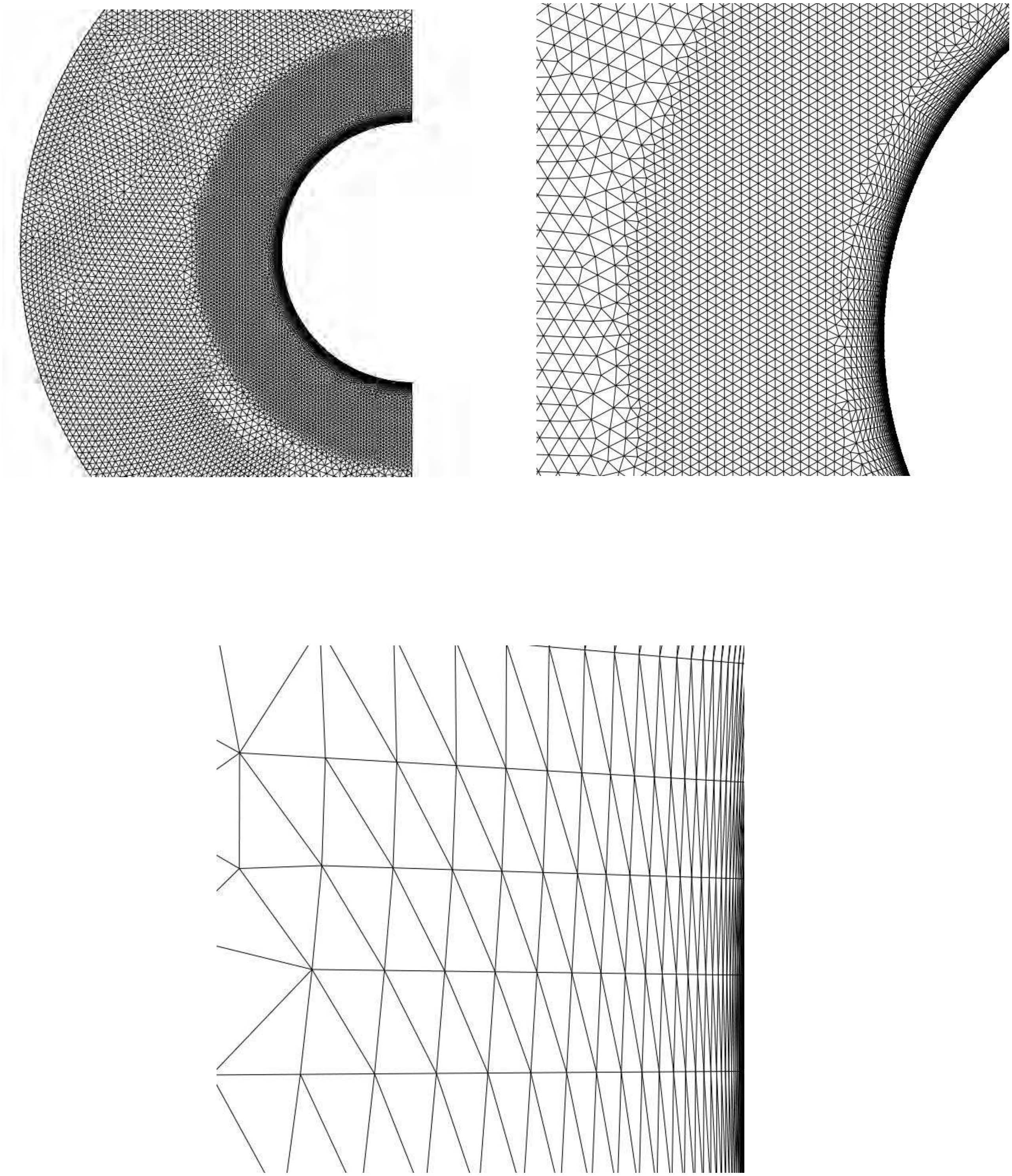
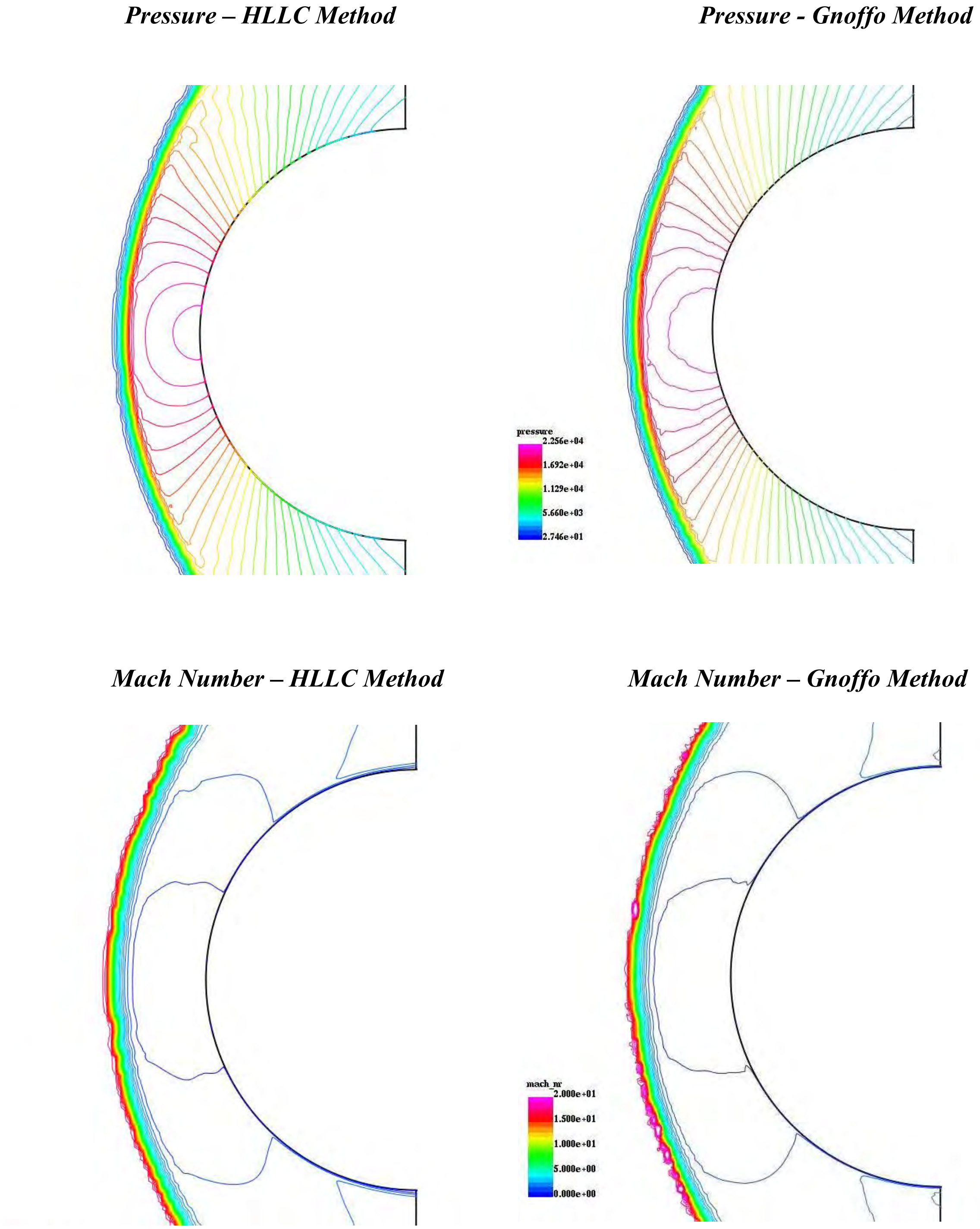


Fig. 4.2 Blunt Body Mesh - Full, Close-up, Zoom



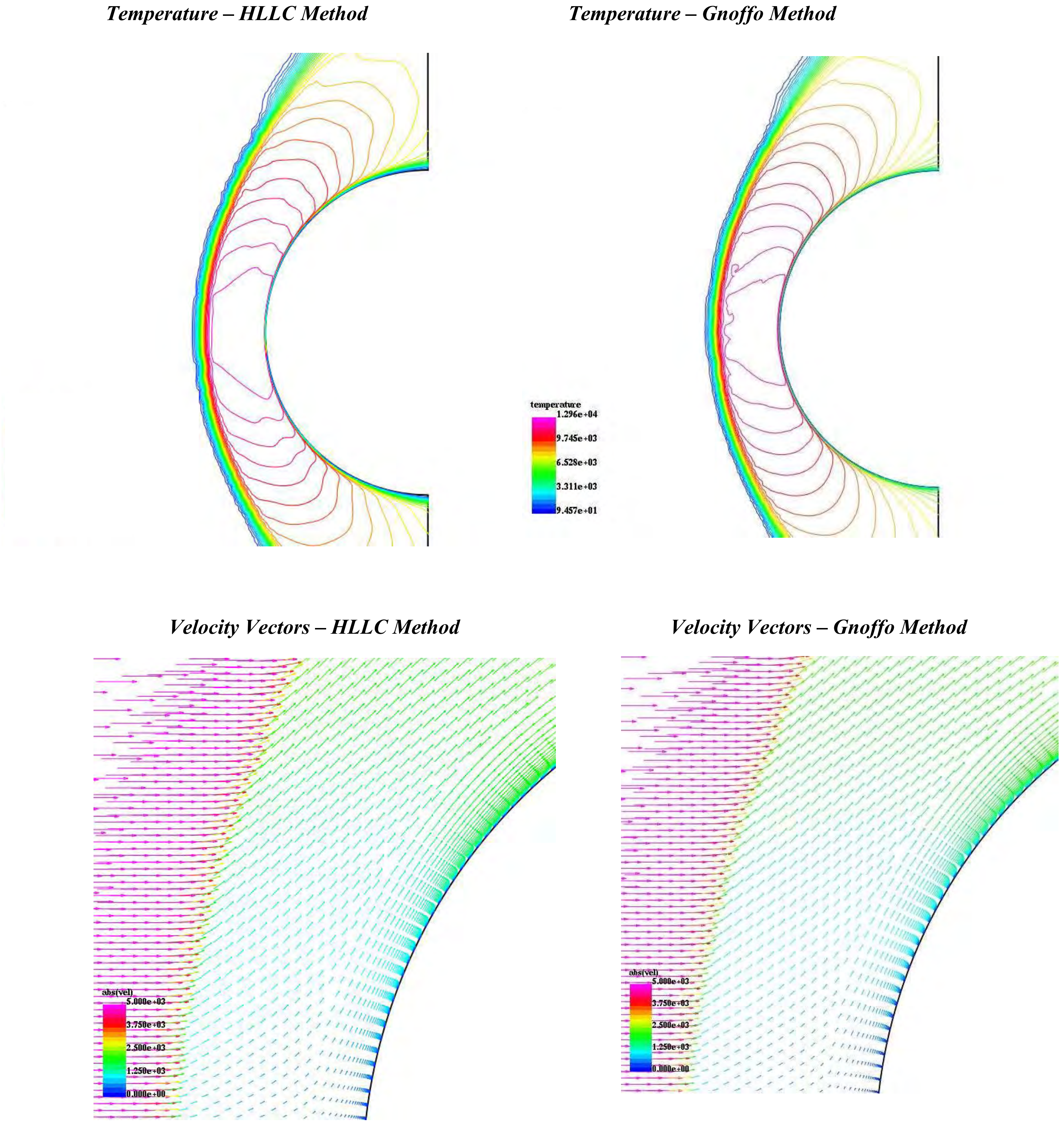
Pressure – HLLC Method

Pressure - Gnoffo Method

Mach Number – HLLC Method

Mach Number – Gnoffo Method

Fig. 4.3 Pressure and Mach Number Comparisons


Temperature – HLLC Method

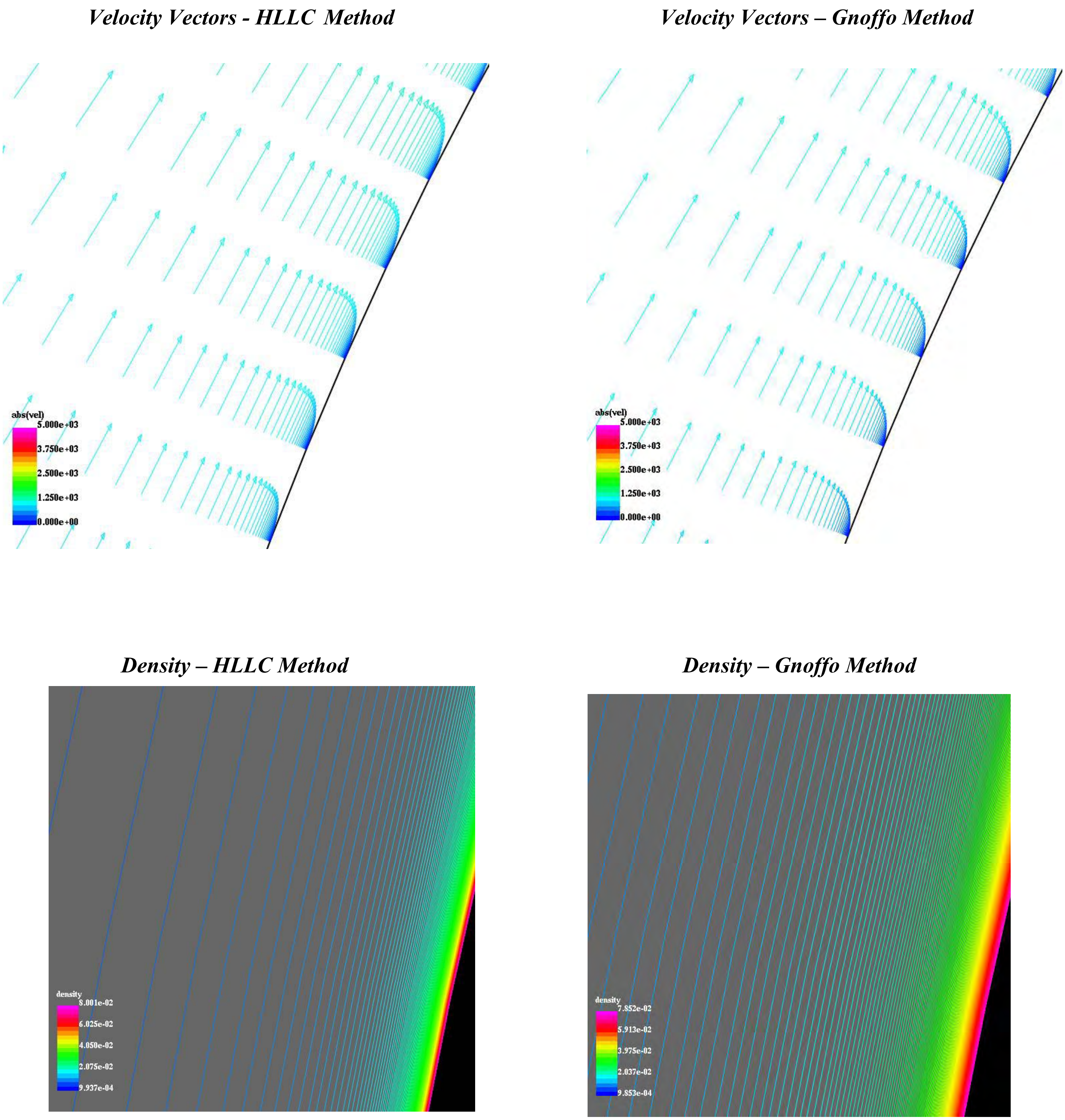
Temperature – Gnoffo Method

Velocity Vectors – HLLC Method

Velocity Vectors – Gnoffo Method

Fig. 4.4 Temperature and Velocity Vectors Comparisons

.



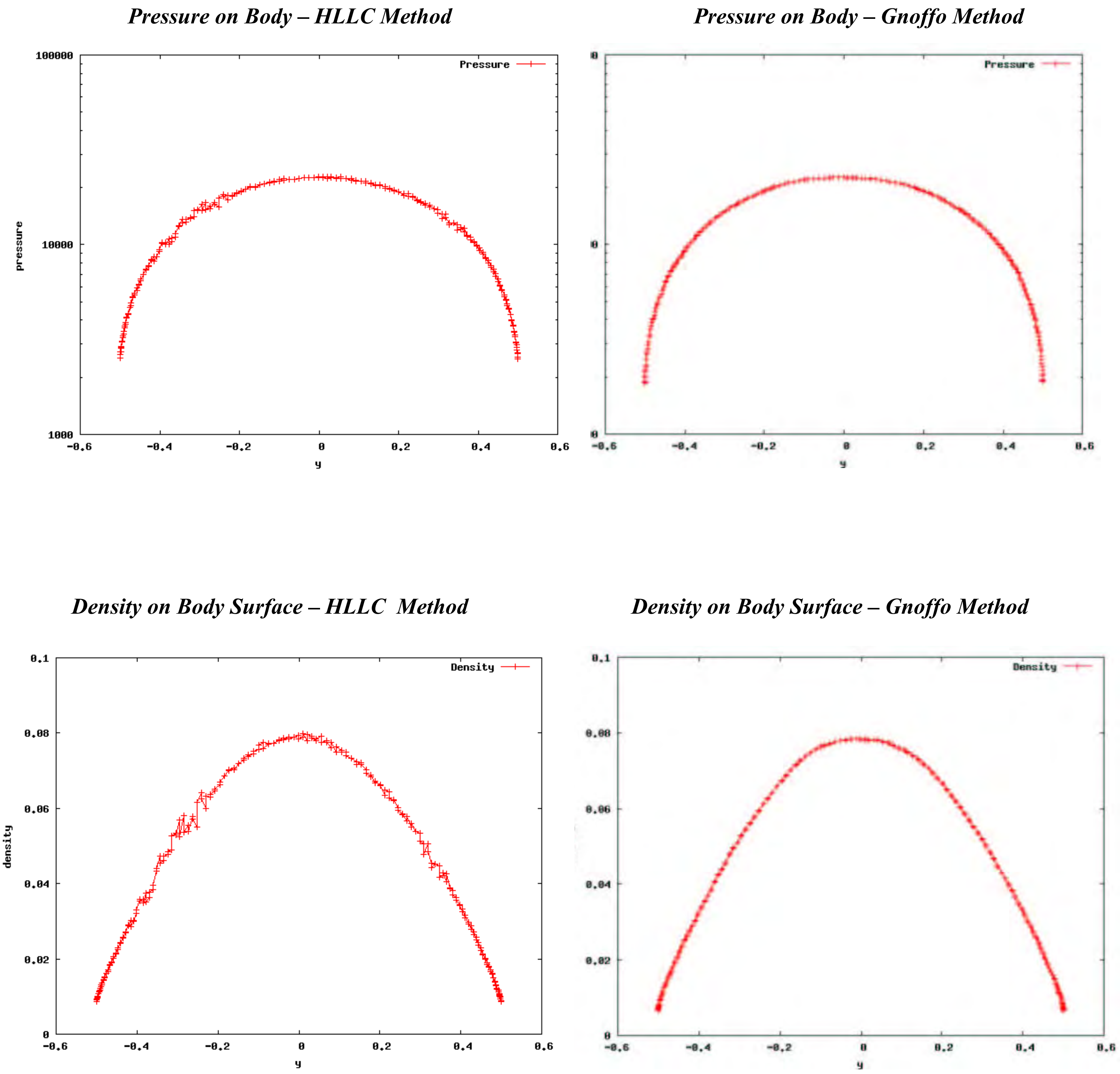
Velocity Vectors - HLLC Method

Velocity Vectors – Gnoffo Method

Density – HLLC Method

Density – Gnoffo Method

Fig. 4.5 Zoomed Views of Velocity Vectors and Density near Body



Pressure on Body – HLLC Method

Pressure on Body – Gnoffo Method

Density on Body Surface – HLLC Method

Density on Body Surface – Gnoffo Method

Fig. 4.6 Pressure and Density on Body Surface

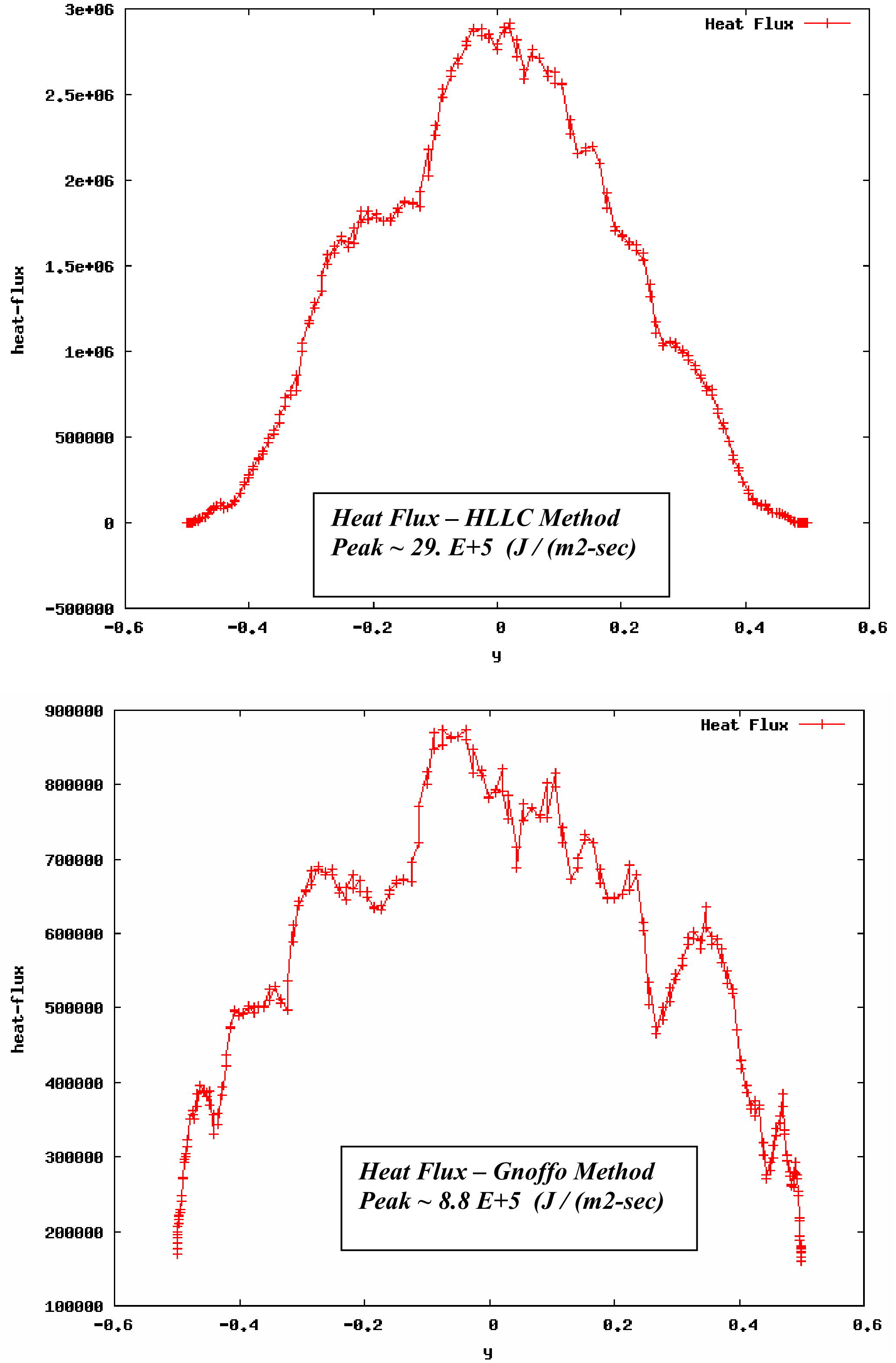


Fig. 4.7 Comparison of Heat Flux

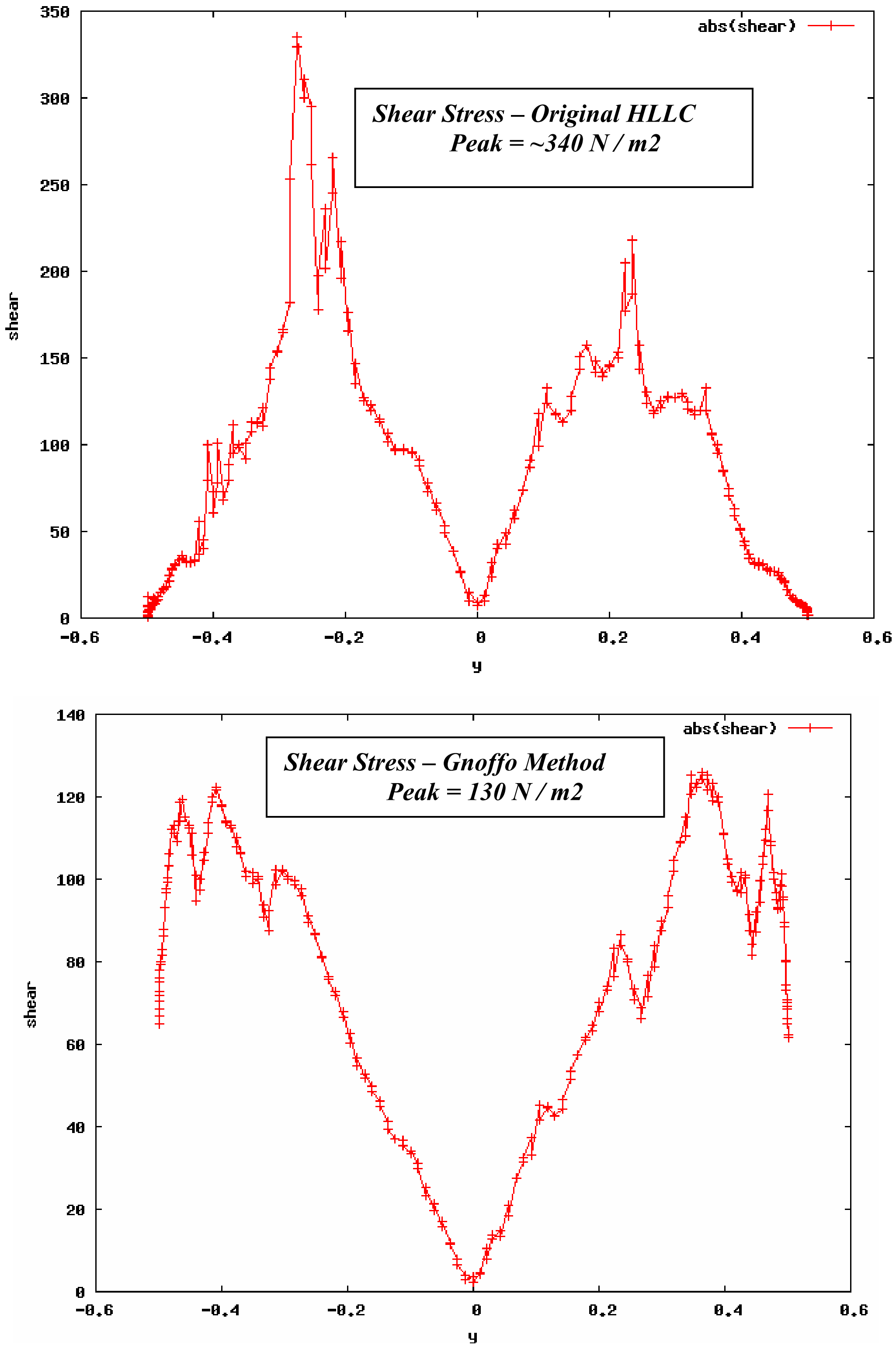


Fig. 4.8 Comparison of Abs Shear Stress

4.4 Recommendations

In light of the many years of research into the aforementioned hypersonic heating problem, we recommend research using a different approach. This different methodology starts from a kinetic formulation based on extensions of the Lattice Boltzmann equation [6]. The idea of this approach is that it is simpler to model complex physics first at the kinetic level, and then use coarse-graining procedures only after modeling is done. The appropriately modeled kinetic equations are solved numerically, and then the dynamic variables are averaged to obtain coarse-grained results for macroscopic quantities such as velocity, pressure, shear stress, heat transfer, and aerodynamic forces and moments.

The Lattice Boltzmann approach is receiving much attention in the international literature [7, 8 9, 10]. Ubertini and Succi [7] published a report on use of Lattice Boltzmann method in conjunction with a conventional finite volume formulation. Their method improves stability and accuracy over the standard finite volume method without a run time penalty. Another review of this approach is given by Shu and associates [8], from Singapore. They show results for micro flows, multi-phase, high Reynolds number viscous simulations and the first results for a problem with a strong shock wave.

A recent paper by Ji and associates [9], also from Singapore, shows initial results for a finite volume method in which the Godunov flux solver is replaced by a Lattice Boltzmann flux solver. The advantage is that a full three-dimensional Riemann problem is solved for the cell face fluxes rather than an approximate one-dimensional solution. Their results showed improved shock resolution in a shock tube computation over the conventional Godunov flux. These authors conclude that the Lattice Boltzmann flux solver has a potential to be an efficient method for compressible flows.

In search for a better finite volume method flux solver, we recommend an investigation to establish proof-of-concept for a Lattice Boltzmann method for constructing the inviscid fluxes in conventional finite volume methods. Our recommendation has the potential for resolving many of the problems in current finite volume methods for hypersonic turbulent flows.

The development will be done with our own finite volume software FEFLO, however all routines will use a totally modular concept for direct inclusion into NASA codes. Our approach will consist of developing the Lattice Boltzmann flux solver and implementing it into FEFLO. We will then define and compute, with NASA coordination, a hypersonic flow for a blunt body. We will also run the case using conventional flux methods such as Roe and HLLC, and the Gnoffo method [5] and make comparisons. If the Lattice Boltzmann approach proves to be a significant improvement, we will complete the development, provide a library of solvers, add our anisotropic adaptive meshing, add the Lattice Boltzmann module to a NASA code of choice, run extensive hypersonic turbulent cases, compare to NASA data bases, document and deliver the modules.

4.5 References

[1] Spradley, Lawrence W. and Rainald Lohner, "Multidisciplinary Software for Analysis and Design of HRRLS Vehicles", AFOSR / NASA Foundational Hypersonics Program Review, Virginia Beach, VA , 5-9 October 2009.

[2] Gnoffo, Peter A, "Multi-Dimensional, Inviscid Flux Reconstruction for Simulation of Hypersonic Heating on Tetrahedral Grids", AIAA 2009-599, Orlando, FL., Jan 2009.

[3] Chandler, G, "Advances in the Computation of Hypersonic Flows", AFOSR / NASA Foundational Hypersonics Program Review, Virginia Beach, VA, 5-9 October 2009.

[4] Holden, M, "CUBRC – Experimental Characterization of Hypersonic Flows, AFOSR / NASA Foundational Hypersonics Program Review, Virginia Beach, VA, 5-9 October 2009.

[5] Gnoffo, Peter, A. "Updates to Multi-Dimensional Flux Reconstruction for Hypersonic Simulations on Tetrahedral Grids", AIAA 2010-1271, Orlando, FL, Jan 2010.

[6] Spradley, Lawrence W. and S. James Robertson, "Kinetic Boltzmann Turbulence Method for Improved Predictions of High Energy Beam Quality", ResearchSouth, Inc., MDA SBIR 04-020, Missile Defense Agency, November 2004.

[7] Ubertini, Stefano, and Sauro Succi, "A Generalised Lattice Boltzmann Equation on Unstructured Grids", Communications in Computational Physics, Vol. 3, No. 2, pp 342-356, 2008.

[8] Shu, C., et al, "Progress in the Development and Application of Lattice Boltzmann Method", Proc. 12th Asian Congress of Fluid Mechanics, Daejeon Korea, 2008.

[9] Ji, C.Z., C. Shu, and N. Zhao, " A Lattice Boltzmann Method – Based Flux Solver and Its Application to Solve Shock tube Problem, Modern Physics Letters, Vol 23, No. 3, pp 313-316, 2009.

5.0 RANS SOLUTION FOR GENERIC HRRLS CONFIGURATION

5.1 Mesh and Flow Conditions

Figure 5.1 shows the generic flyer configuration #2 which has been previously analyzed with an inviscid Euler plus Boundary Layer methodology. The results given here represent a steady state RANS solution for this configuration.

The RANS surface mesh is shown in Fig. 5.2. The total mesh contains about 2.5M tetrahedral elements. This surface mesh and the symmetry plane mesh are shown together in Fig. 5.3. A close-up view on the nose mesh is given in Fig. 5.5. A close up of the tail BL mesh near the surface is shown in Fig. 5.5.

Vertical cut-plane views of the mesh are shown in Figs. 5.6 and 5.7. A cut through the fuselage, about half way to the wing leading edge, is shown in Fig. 5.6. A similar cut, about halfway back on the wing, is shown in Fig. 5.7.

A close up view near the tail region is shown in Fig. 5.8.

.

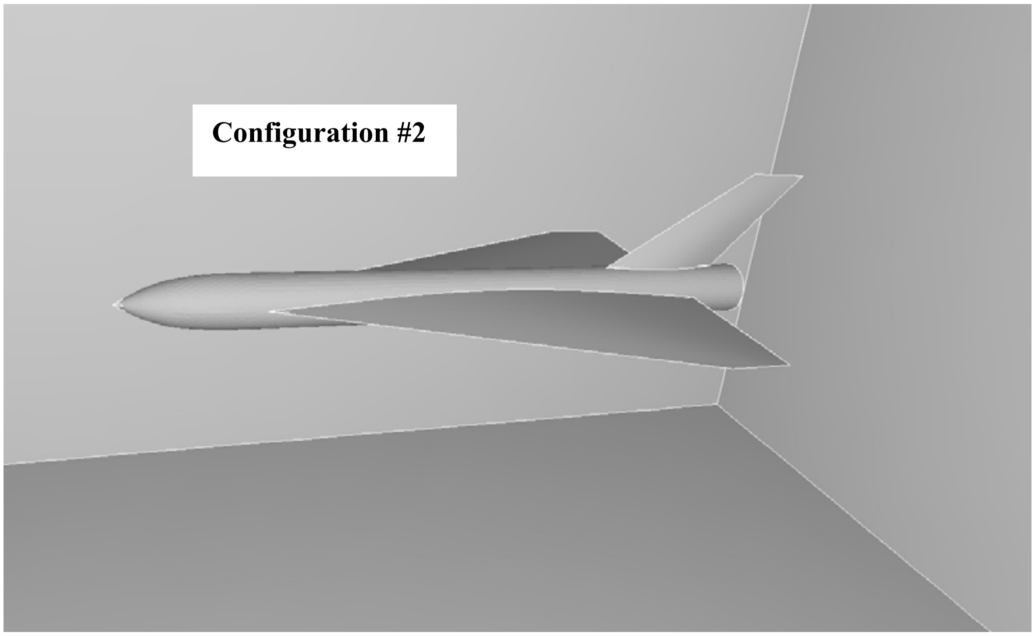


Fig. 5.1 Generic Vehicle Configuration #2

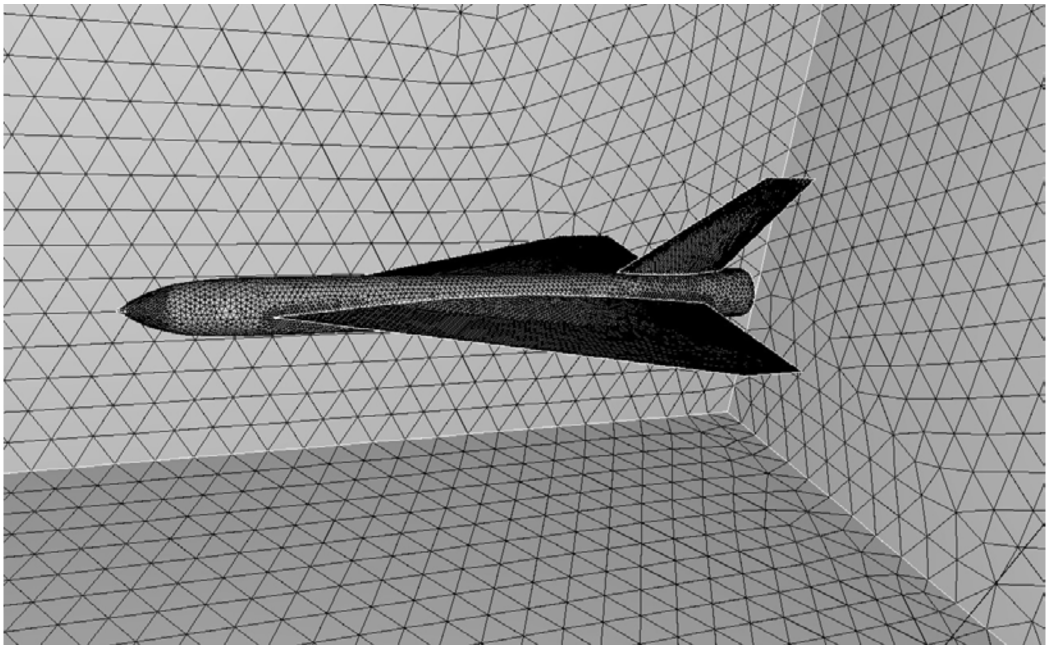


Fig. 5.2 RANS Surface Mesh

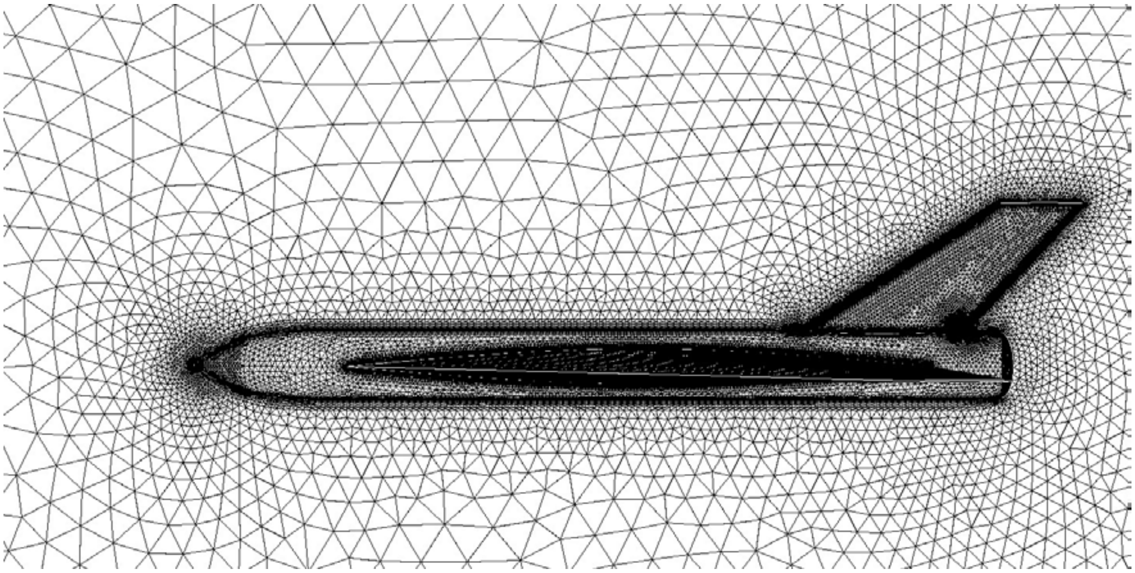


Fig. 5.3 Surface and Symmetry Plane RANS Mesh

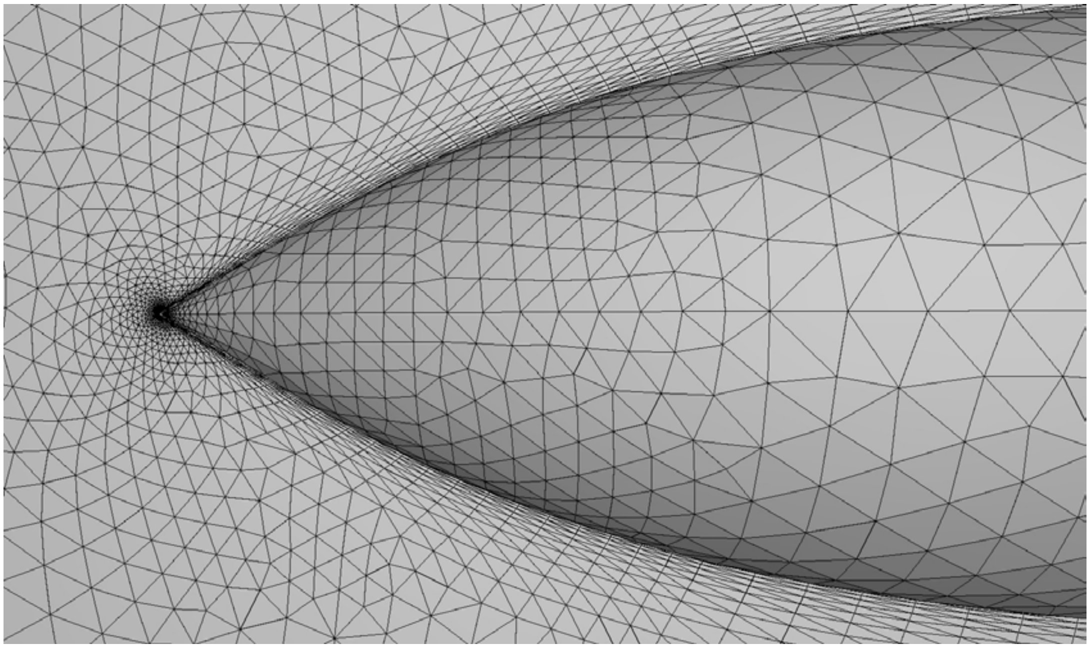


Fig. 5.4 Surface and Symmetry Plane - Nose close up

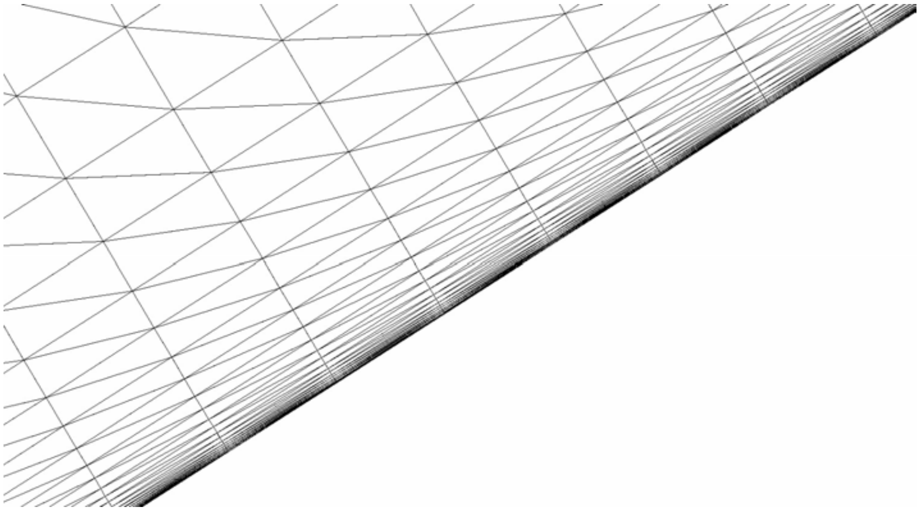


Fig. 5.5 Symmetry Plane Tail Mesh – Close Up

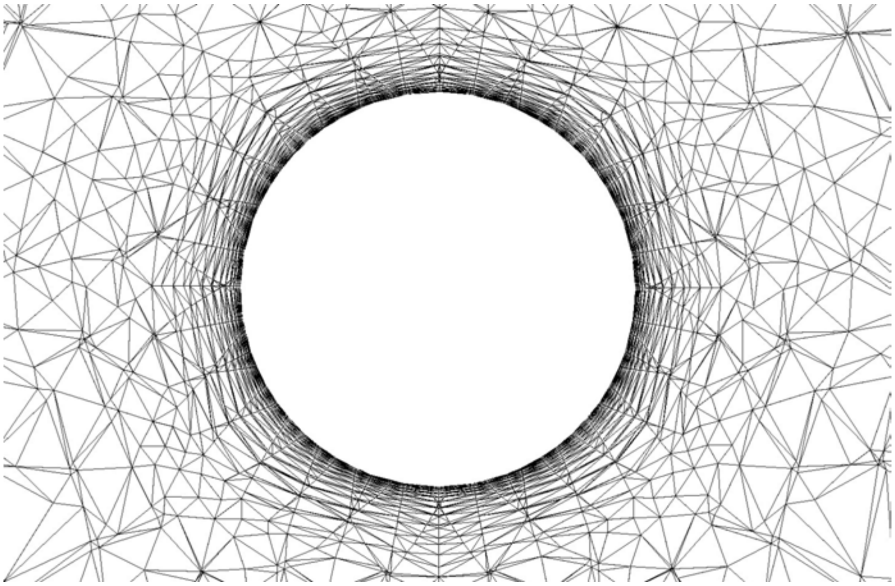


Fig. 5.6 Planar Cut of RANS Mesh thru Fuselage

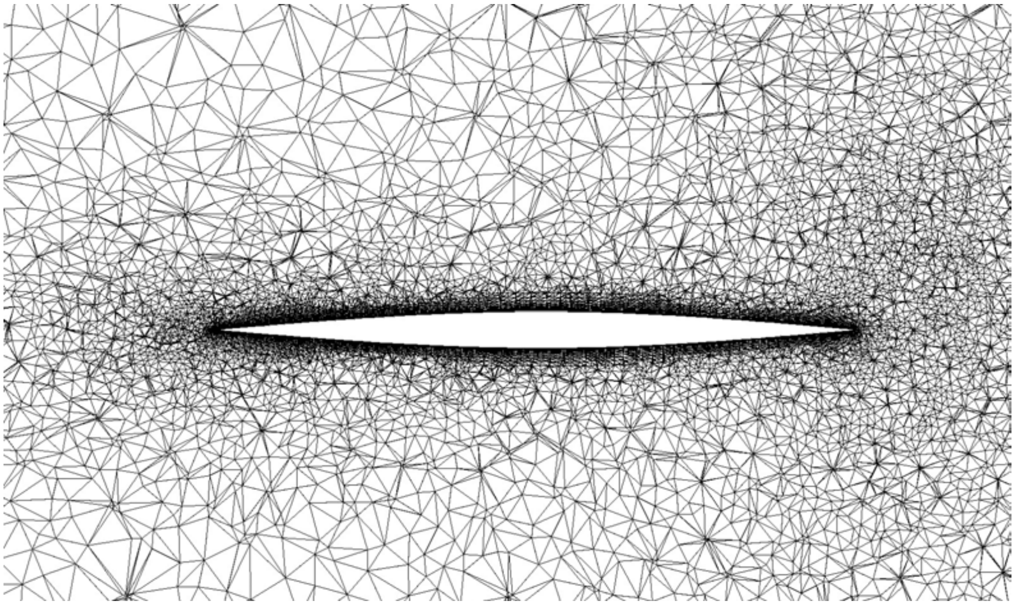


Fig. 5.7 Planar Cut of RANS Mesh thru Wing

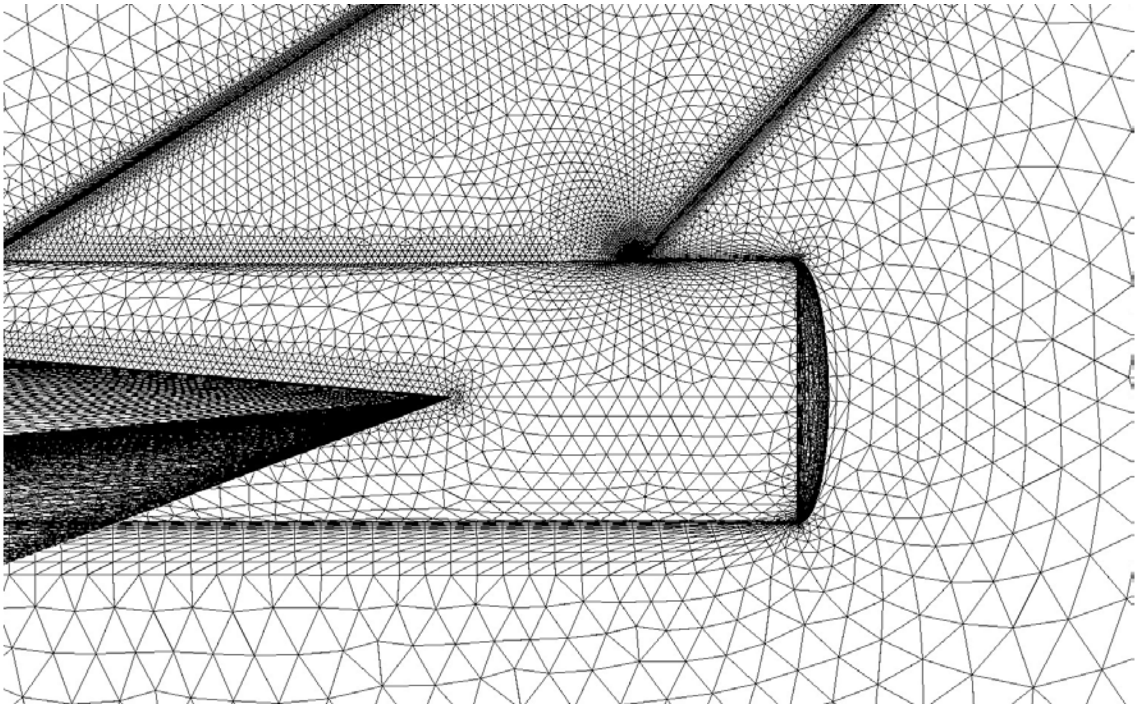


Fig. 5.8 Tail Mesh – Close up

Table 5.1 gives the flow conditions for this case. The flow was assumed to be laminar everywhere at Mach = 5 at 30 km altitude and angle of attack of 5 degrees. The wall temperature was held fixed at 300K. All units are in MKS.

Table 5.1 Flow Conditions (kg / meter/ sec)

density	velocity	pressure	temperature	viscosity	Cp
5.8E-4	*1500. , 130., 0.0*	*43.2*	*261 K*	*1.75E-5*	*1000.*
Length	**Mach**	**Pr**	**Re**	**Tw**	α
20 m	*5.0*	*0.71*	*6.25E+4*	*300*	*5 deg*

5.2 Steady State Solution

The following are steady state solution profiles for the above configuration and flow conditions.

The surface pressure contours for two views are shown in Fig. 5.9. The nose, the wing leading edge and the tail leading edge have the highest surface pressures.

The pressure contours for cut plane views are given in Figs. 5.10 and 5.11. A cut through the fuselage, about half-way between the nose and wing leading edge, shows the high pressure on the windward side. This cut plane plotted with the fuselage is also shown. A cut through the wing, about midway from the leading edge, shows the higher pressure at the wing-body junction.

The plots shown in Figs. 5.12 and 5.13 are contours of absolute velocity with overlays of velocity vectors. These show the boundary layer profiles near the vehicle surface. Figure 5.12 is on the nose portion of the body, and Fig. 5.13 is at mid-fuselage. The Blue contours are lowest velocity and Red is largest.

Gas temperature contours on the nose portion of the body, with velocity vectors, are shown in Fig. 5.14. Again the velocity vectors are overlaid for clarity of the boundary layer. The top figure is a close-up view to show the thermal gradients through the shock layer. An enlargement of this, in the bottom figure, gives details of the thermal boundary layer. The wall is held fixed at 300K.

Figure 5.15 and 5.16 are surface contours of shear and heat flux respectively from the RANS solution.

Plots of surface values plus fuselage cut planes and wing cut planes are next. Figures 5.17, 5.18 and 5.19 show pressure, abs velocity and temperature contours respectively.

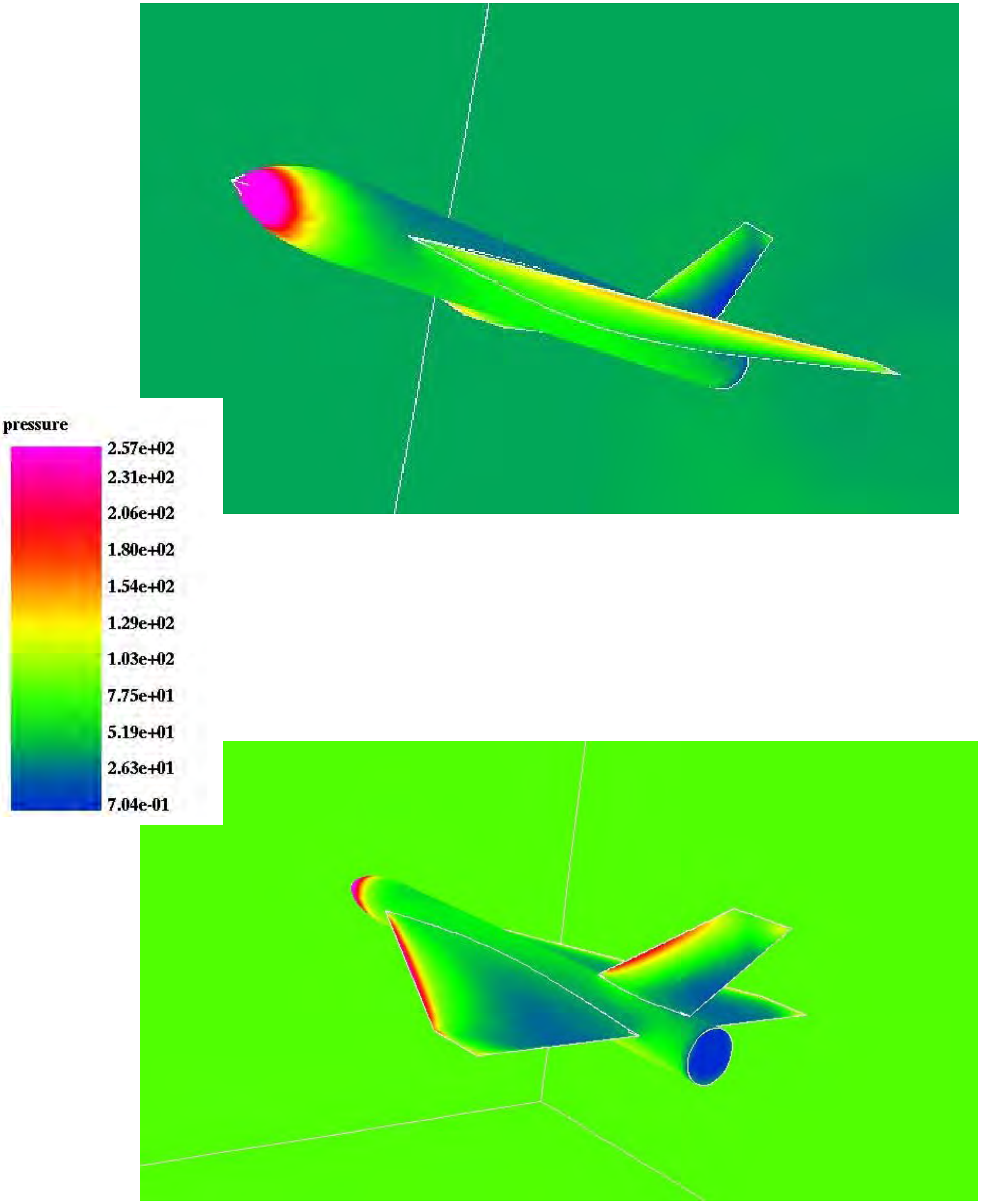


Fig. 5.9 Surface Pressure – Front and Rear Views

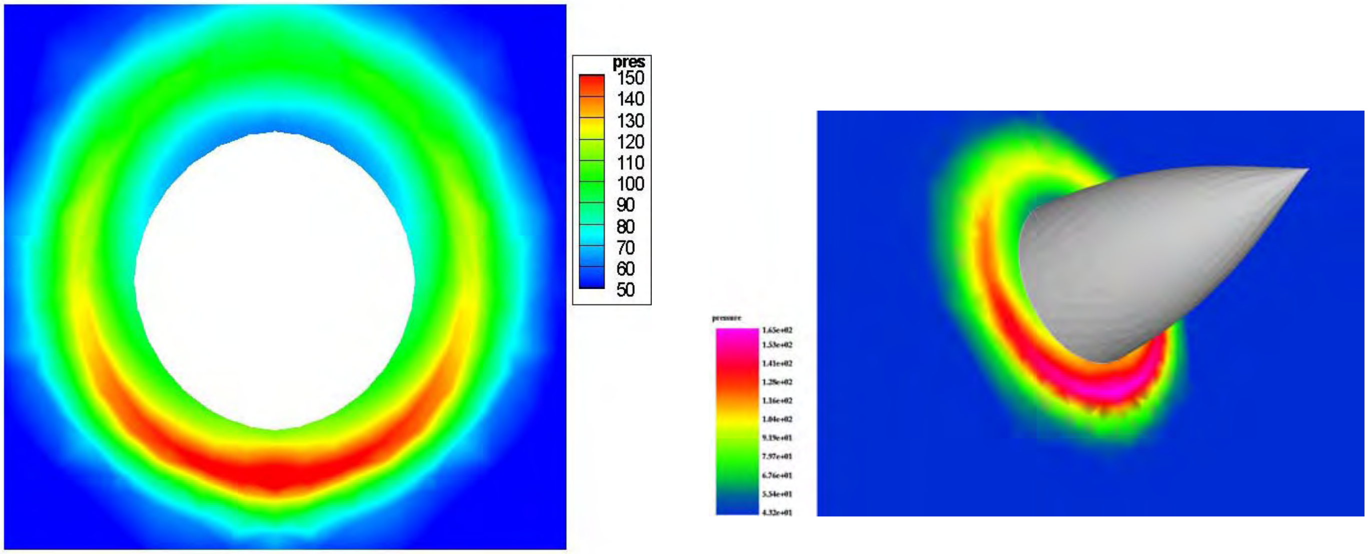


Fig. 5.10 Pressure Contours – Cut Plane thru Fuselage

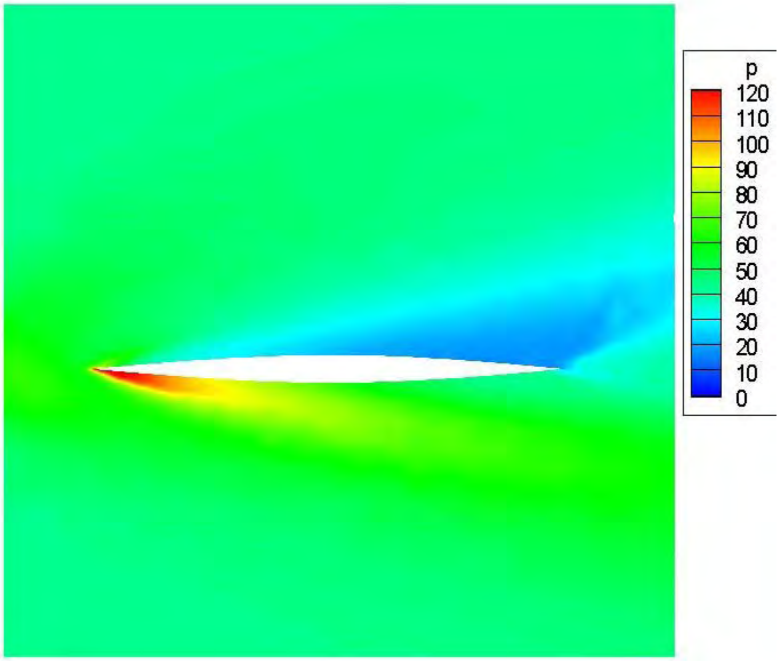


Fig. 5.11 Pressure Contours – Cut Plane through Mid-Wing

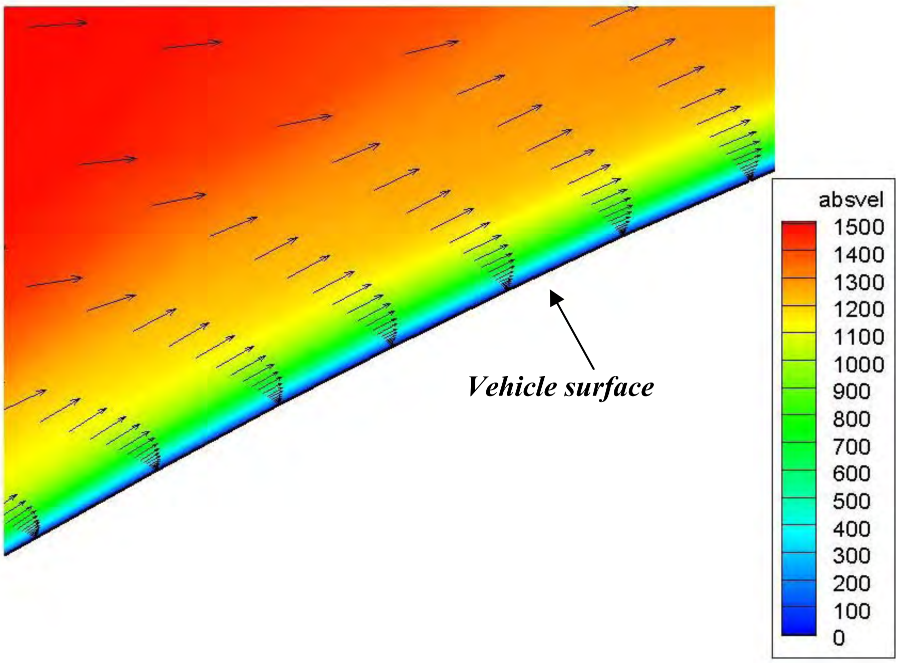


Fig. 5.12 Velocity Vectors and Abs Vel Contours – Nose Region

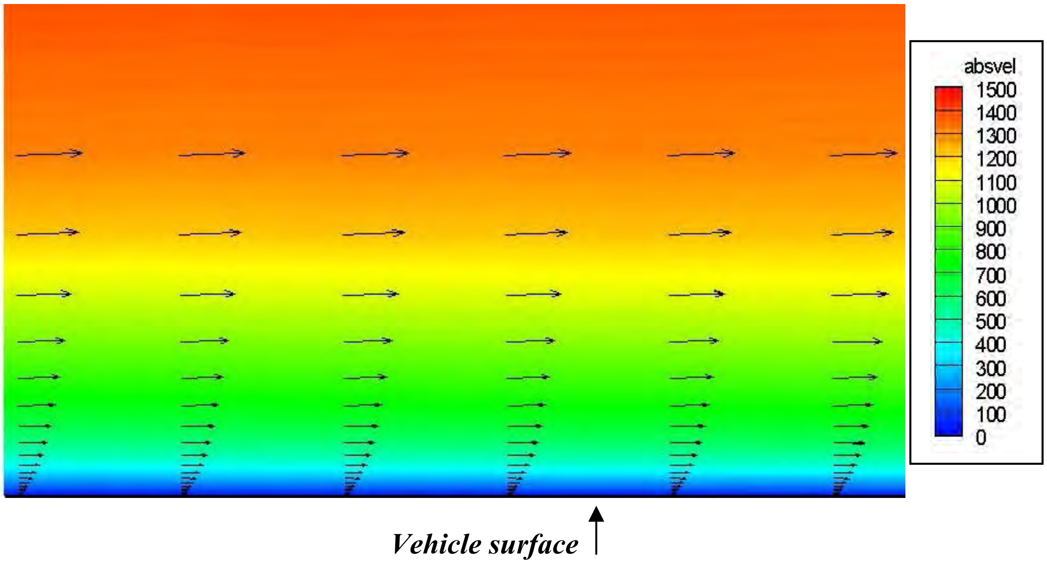
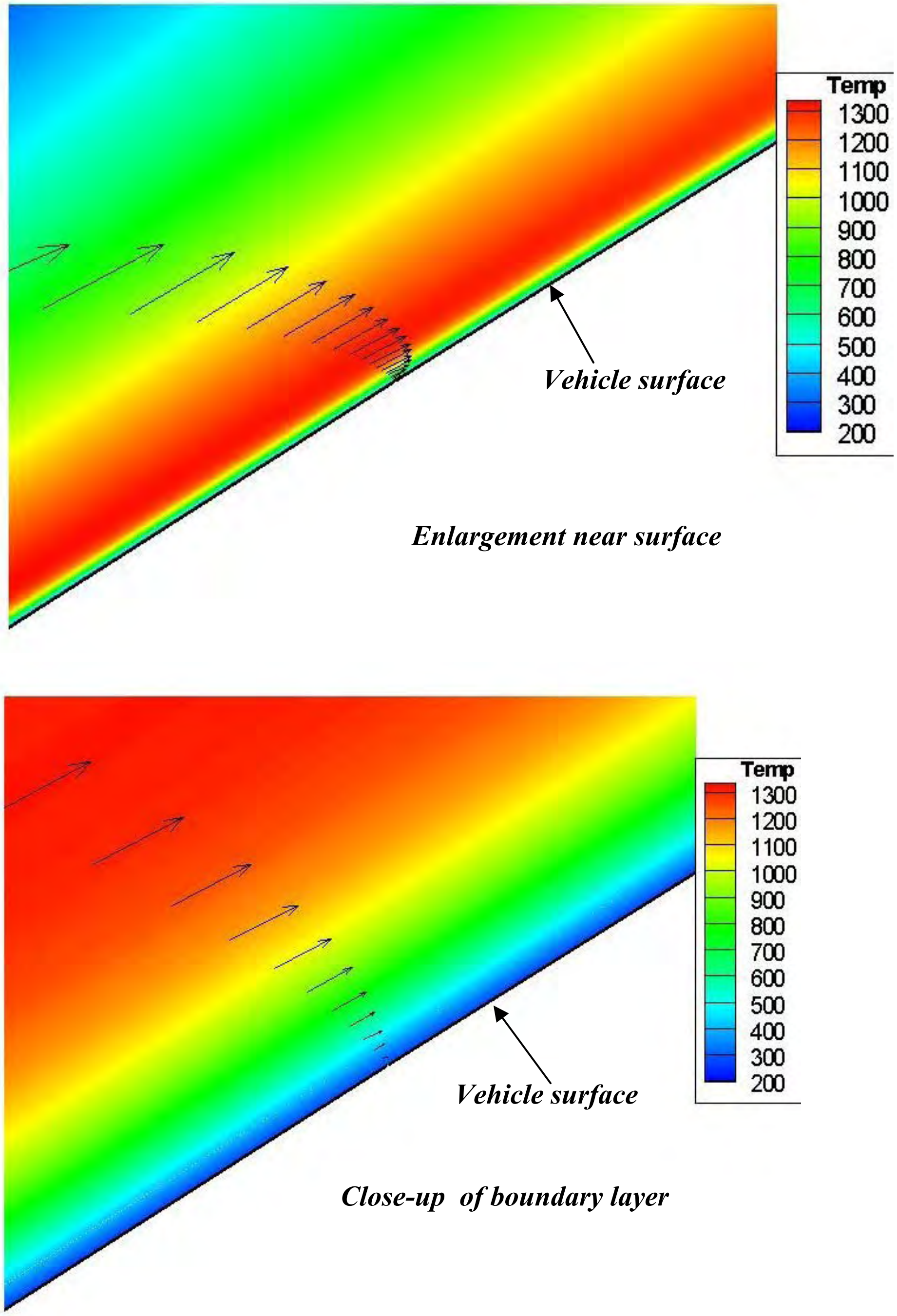


Fig. 5.13 Velocity Vectors and Abs Vel Contours – Mid Fuselage



Enlargement near surface

Close-up of boundary layer

Fig. 5.14 Temperature Contours – Nose Region Overlaid with Velocity Vectors

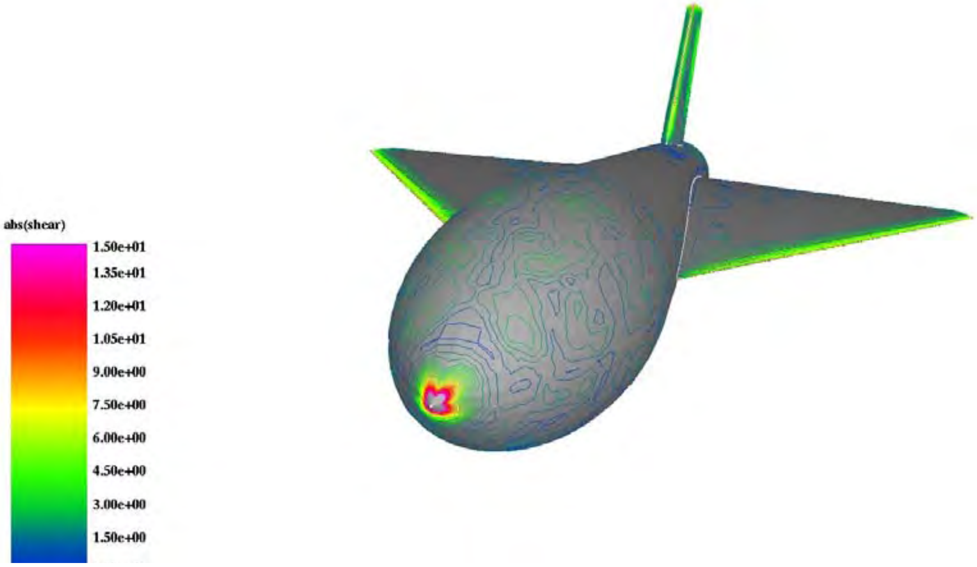


Fig. 5.15 Contours of Abs Shear

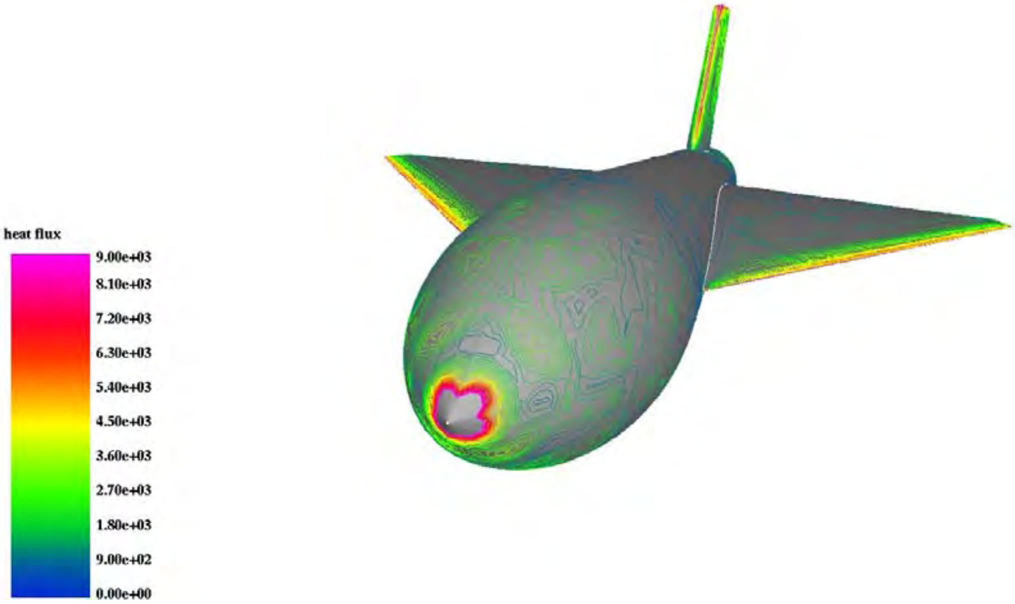


Fig. 5.16 Contours of Abs Heat Flux

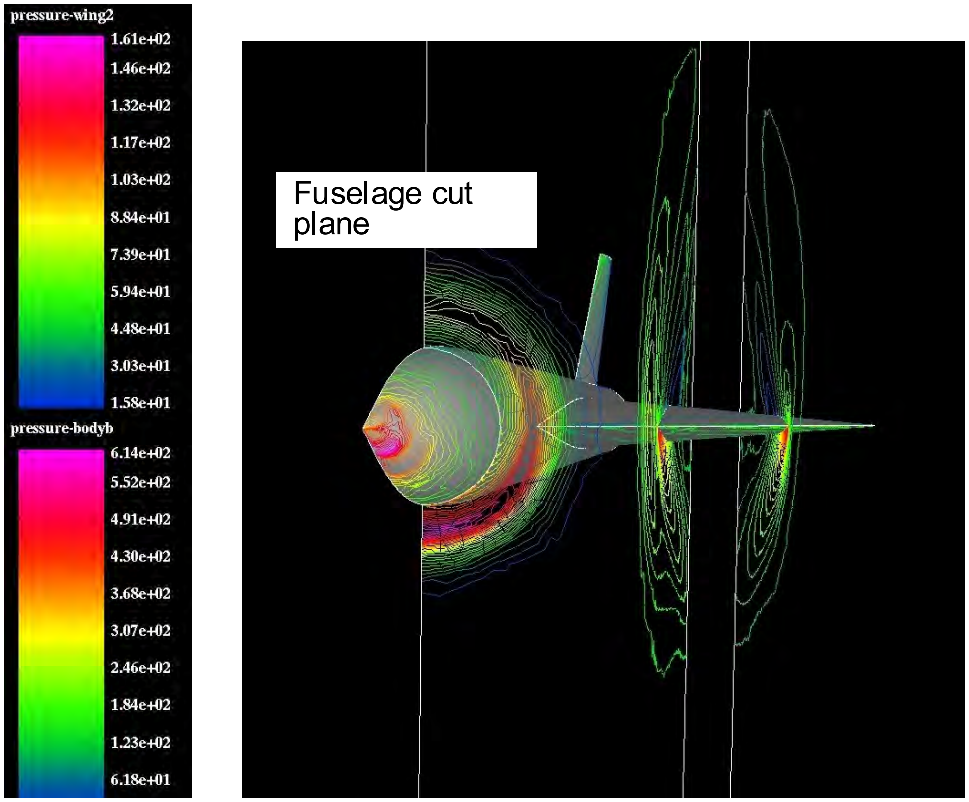


Fig. 5.17 Pressure Contours – Fuselage and Cut Planes

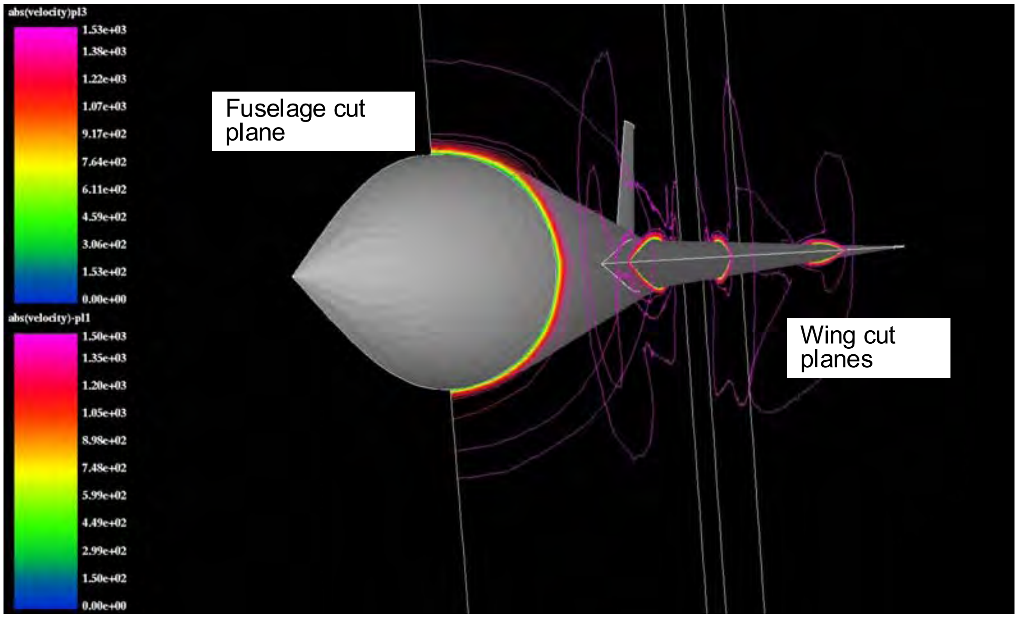


Fig. 5.18 Abs Velocity Contours – Fuselage and Cut Planes

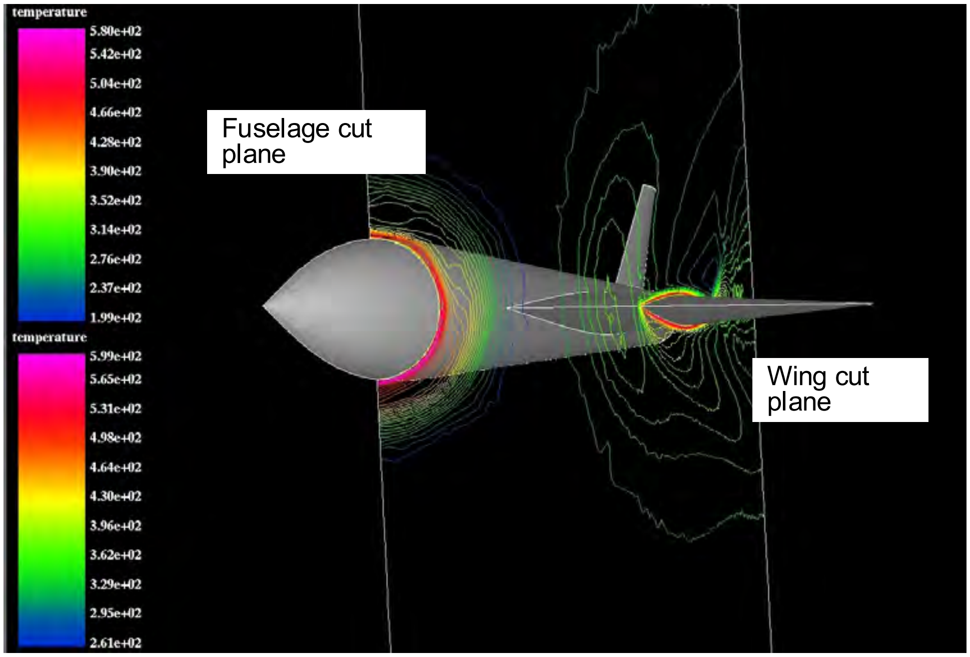


Fig. 5.19 Temperature Contours – Fuselage and Cut Planes

5.3 Recommendations

Hybrid RANS / LES for Multi-Discipline Computation

The current state-of-the-art in viscous flow computation for multi-discipline analysis is Reynolds Averaged Navier Stokes equations (RANS). With the rapid advances in algorithms and cheap / fast computing power now available, we are in a position to move past turbulence modeling in RANS solvers for the large scales of turbulence. Large Eddy Simulations (LES) provide one important move in this direction [1]. The following is a brief summary of a hybrid RANS / LES approach for multi-discipline analysis and design.

Large Eddy Simulation (LES)

LES is a numerical technique used to solve the partial differential equations governing turbulent fluid flow. The basic idea is that of self similarity in that large eddies of the flow are dependent on the flow geometry, while smaller eddies are self similar and have a universal character. It has thus became a practice to solve only for the large eddies explicitly and model the effect of the smaller and more universal eddies on the larger ones. Thus, in LES the large scale motions of the flow are calculated, while the effect of the smaller universal scales (the so called sub-grid scales) are modeled using a sub-grid scale model. The resolved scales are integrated accurately in time using the time-dependent Navier-Stokes equations (RANS). The basics of LES will require two major components: 1) many elements to resolve the relevant viscous scales and 2) high order schemes to accurately propagate the flow vortices over large distances.

The main advantage of LES over RANS approaches is the increased level of detail. While RANS methods provide averaged results, LES is able to predict instantaneous flow characteristics and resolve turbulent flow structures. This is particularly important in simulations involving chemical reactions, such as the combustion of fuel in an engine. While the averaged concentration of chemical species may be too low to trigger a reaction, instantaneously there can be localized areas of high concentration in which reactions will occur. LES also offers significantly more accurate results over RANS for flows involving flow separation or acoustic prediction.

Progress has been made in LES primarily on structured Cartesian meshes. The reasons for this include the fact that LES requires at least fourth order accuracy. This is easy to achieve on simple x-y-z meshes, but more difficult on unstructured tetrahedral meshes. Thus, we gain an advantage while giving up another. Our idea is to use general hexahedral cells to develop a high order stencil in much the same way as in Cartesian coordinates while retaining the unstructured meshing for complex geometries.

LES requires less computational effort than direct numerical simulation (DNS) but more effort than those methods that solve the Reynolds-averaged Navier-Stokes equations (RANS). Zonal approaches are often used, with RANS replacing LES in the wall region. This is the basis of our recommended hybrid approach.

Hybrid Methods

Reference 2 gives a review and assessment of the current state of hybrid LES methods.

A new method in which non-slip conditions are still applied at the wall and a hybrid type approach to turbulent flow simulation was attempted. The papers that they review, proposed using Reynolds stress models in the near wall flow and a smoothing function to scale the eddy viscosity by a parameter α to match with that of outer flow obtained from LES. Another idea was further promoted and referred to as Limited Numerical Scales (LNS) simulation. They used non-linear k - ε RANS model near the walls. A latency factor α is expressed as a function of local turbulent length and velocity scales.

Further literature reviews present a model, which is a one-equation RANS model, which solves one transport equation to calculate eddy viscosity. The destruction term in the transport equation is inversely proportional to length scale using the distance to the nearest wall. If this length scale is redefined suitably based on the local grid, then the eddy viscosity crosses over from the usual Spalart–Allmaras RANS eddy viscosity near the wall to proposed LES eddy viscosity, similar to that of Smagorinsky, away from the wall. This approach is often referred to as 'Detached Eddy Simulation' (DES) since it is intended to be used in regions such as separated regions, in which only eddies that are detached from the surface must be resolved for accurate LES of the flow away from the walls.

Following reference 2, we are now performing R & D at ResearchSouth and George Mason University on a hybrid approach. It is constructed by combining various other RANS models with SGS models. In general, if the same turbulence model is used to obtain both RANS eddy viscosity as well as SGS viscosity, we refer to the method as DES and otherwise as Hybrid LES.

This hybrid LES method for computing turbulent flows is an excellent candidate for extending the state–of–the-art in multidiscipline analysis and design.

5.4 References

[1] R. Lohner, et al, "Large-Eddy Simulation of a Supersonic Jet and its Near-Field Acoustic Properties: Methodology and Validation", *AIAA-09-0500 (2009).*

[2] Vengadesan, S. and P. Nithiarasn, "Hybrid LES – Review and Assessment", Sadhana, Vol. 32, Part 5, Oct 2007, pp. 501-511, India.

6.0 RANS FLOW INTEGRATION TO STRUCTURES FEA

6.1 Steady State Solution

Meshes

Figure 6.1 shows the flyer configuration #2 which has been analyzed in the previous section. The results given here represent a RANS solution for this configuration with loose coupling to the finite element solver FEAP.

The RANS and FEAP surface meshes are shown in Fig. 6.2. The total volume mesh contains about 2.7M tetrahedral elements. The CSD mesh (FEAP) surface mesh contains about 6.3K quad elements. This particular set up does not include internal structure.

Flow Conditions

Table 6.1 gives the flow conditions for this case. The flow was assumed to be laminar everywhere at Mach = 5 at 30 km altitude and angle of attack of 5 degrees. The wall temperature was held fixed at 300K. All units are in MKS.

Table 6.1 Flow Conditions (kg / meter/ sec)

density	**velocity**	**pressure**	**temperature**	**viscosity**	**Cp**
6.8E-4	***1500. , 130., 0.0***	***43.2***	***261 K***	***1.75E-5***	***1000.***
Length	**Mach**	**Pr**	**Re**	**Tw**	α
20 m	***6.0***	***0.71***	***6.25E+4***	***300***	***5 deg***

Steady Solution - RANS to FEAP

The following are steady state solution profiles for the above configuration and flow conditions.

Figure 6.3 shows the RANS surface mesh in the top portion and the steady state surface pressure contours in the lower part. These pressure loads were passed to FEAP where the stresses and displacements were computed. In addition, the viscous and thermal loads, as computed in the RANS solution, were also passed to FEAP. The shear and heat flux contours are plotted in Fig. 6.4.

The loosely – coupled solution reached steady state in 5 passes. The resulting displacement contours and displacement vectors are plotted in Fig. 6.5. These are colored on velocity magnitude and indicate the direction of the displacement. Note these are for the steady state case.

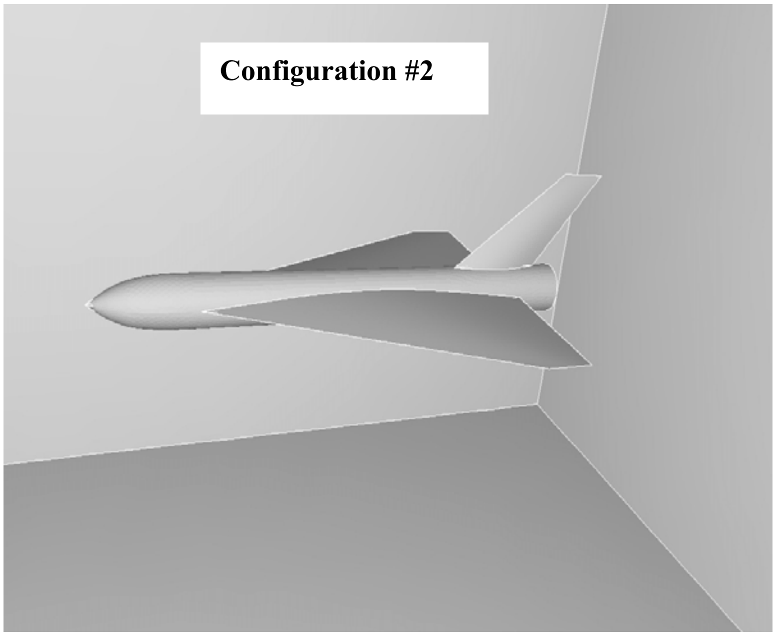


Fig. 6.1 Generic Configuration #2 for RANS Coupling to FEAP

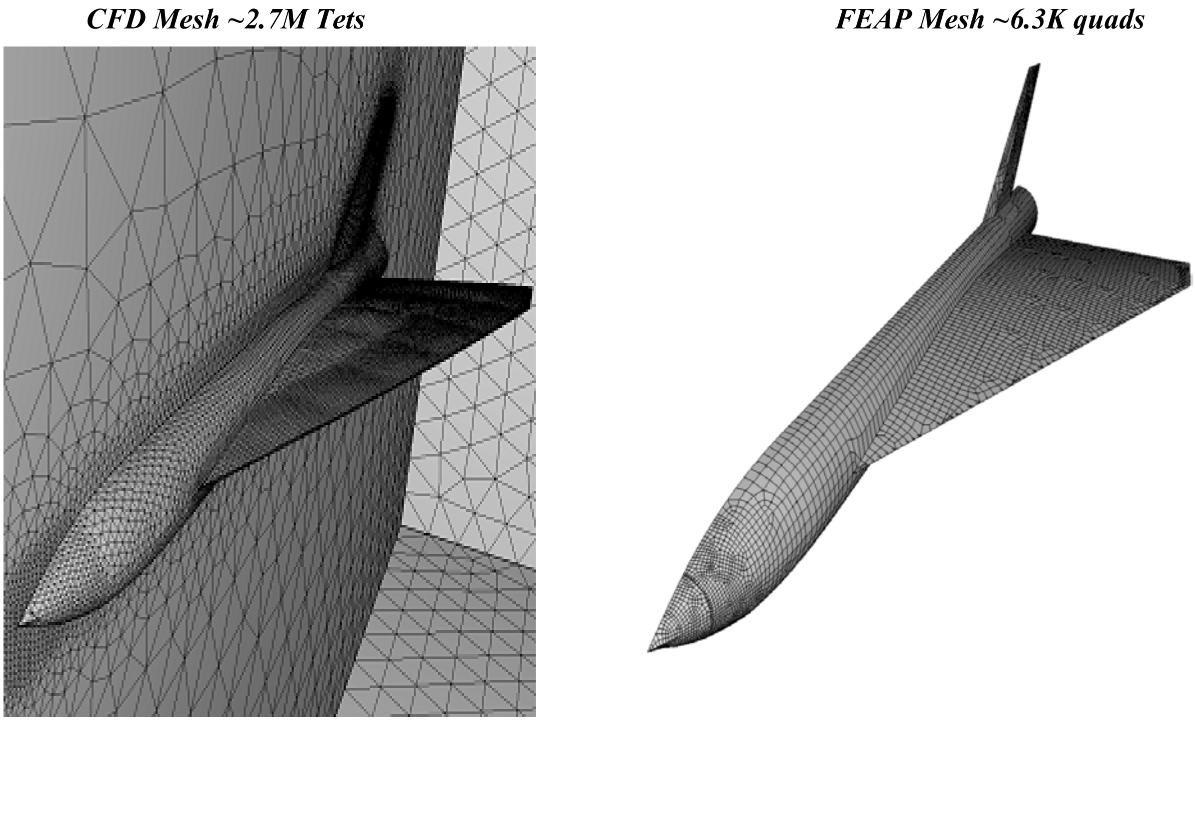
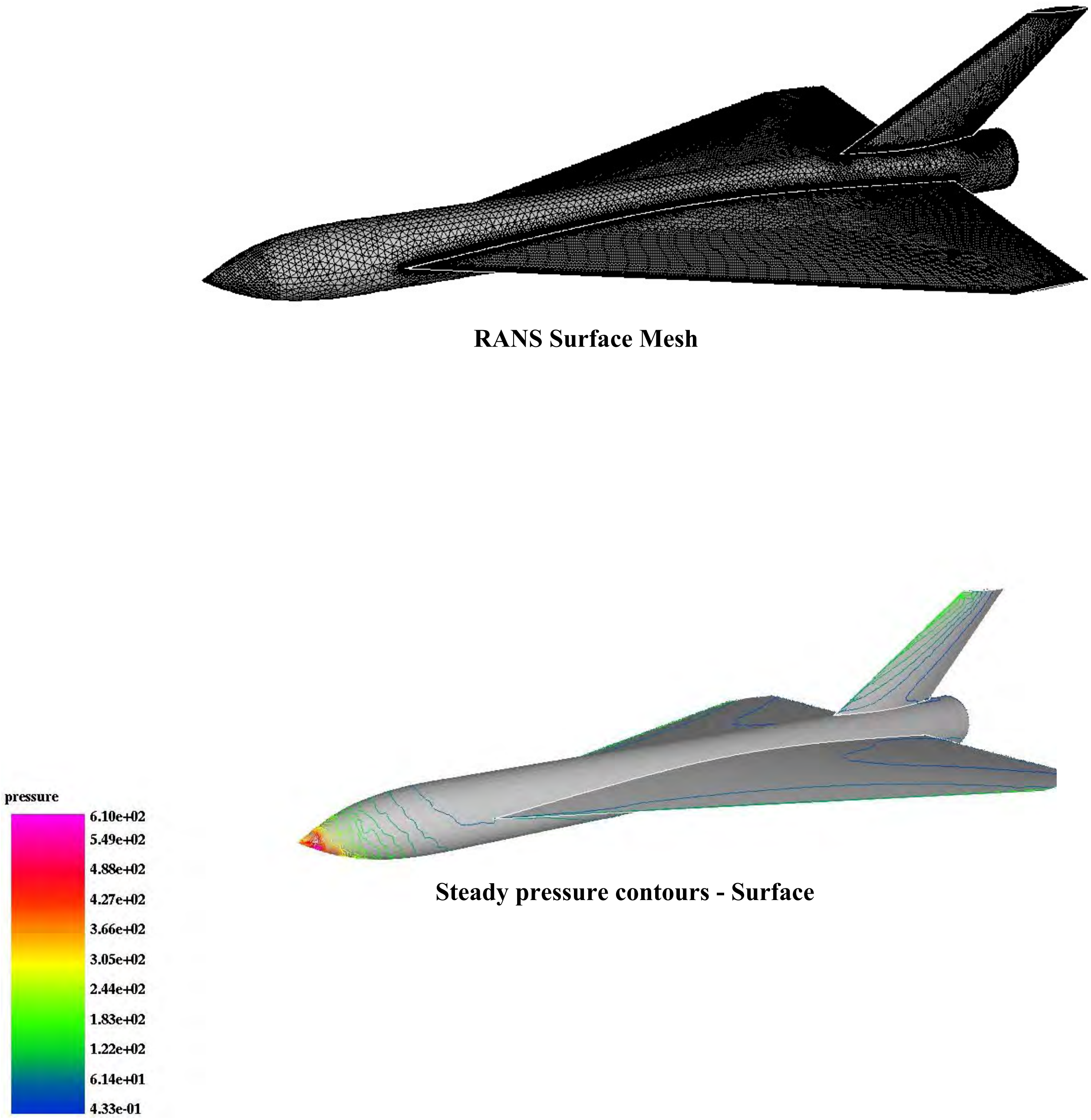


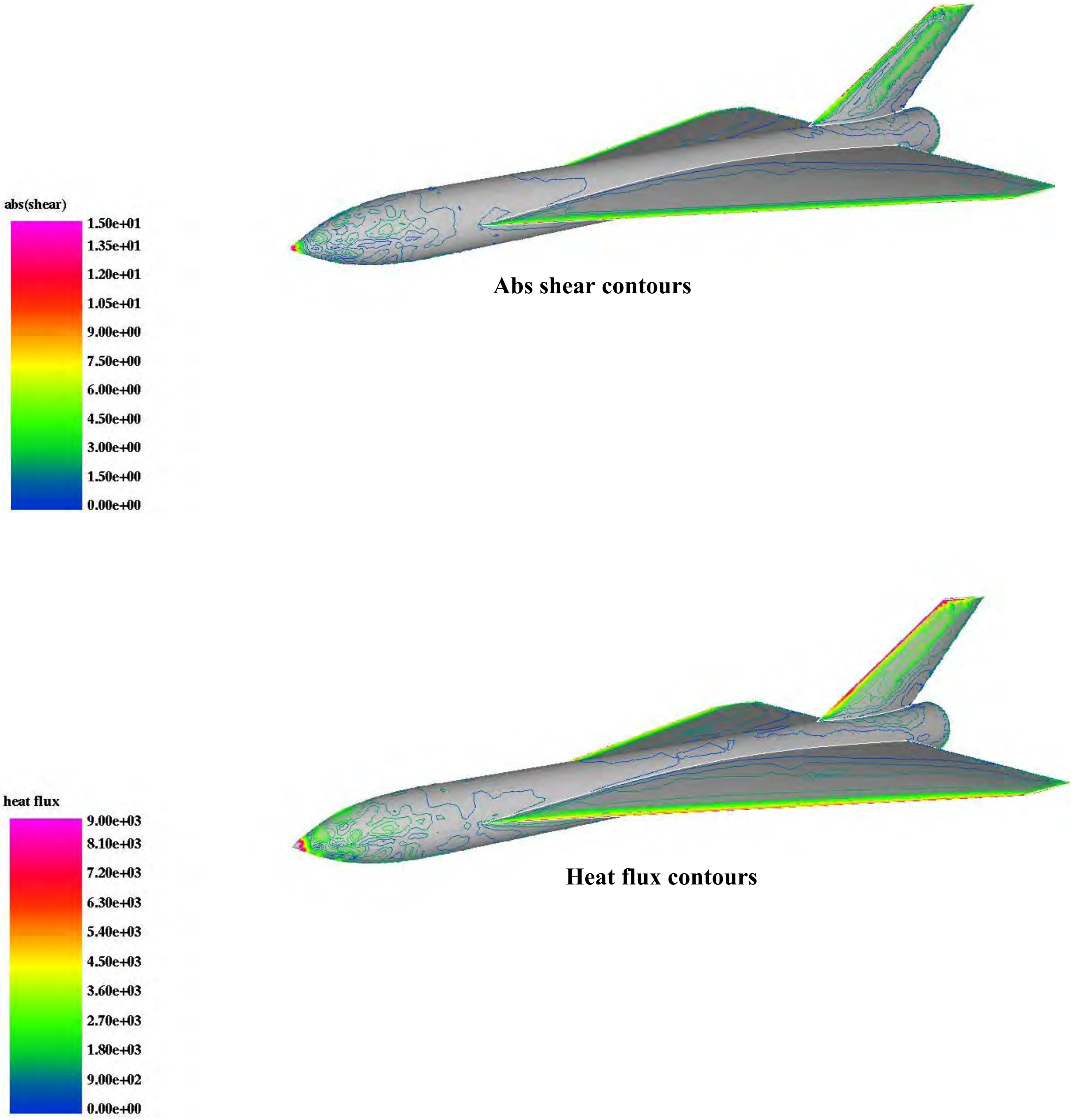
Fig. 6.2 RANS Symmetry Plane Mesh and FEAP Surface Mesh



RANS Surface Mesh

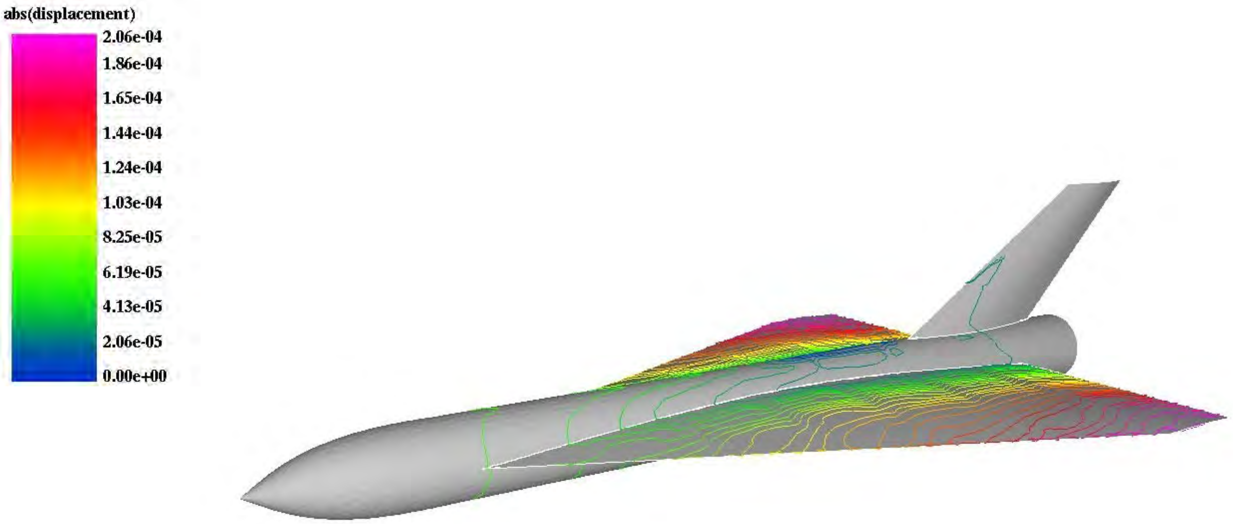
Steady pressure contours - Surface

Fig. 6.3 RANS Surface Mesh and Steady Pressure Contours

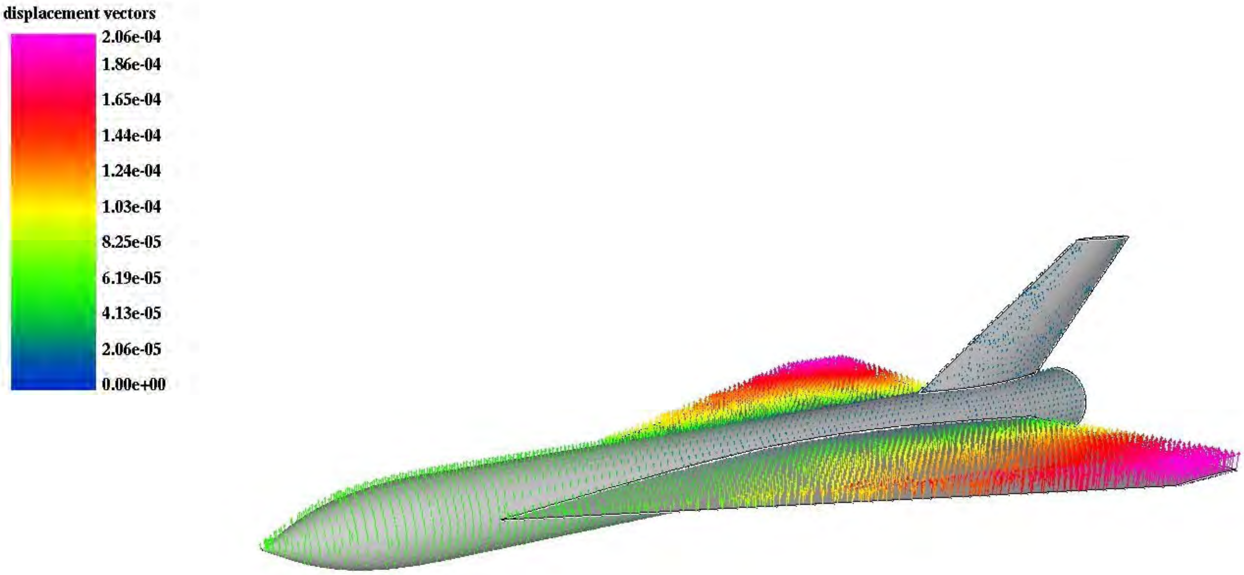

Abs shear contours

Heat flux contours

Fig. 6.4 Absolute Shear and Heat Flux Contours – Steady State



Abs displacement contours



Displacement vectors
Colored on velocity magnitude

Fig. 6.5 Absolute Displacement Contours and Displacement Vectors

6.2 Unsteady Solution

Meshes

Figure 6.6 shows hypersonic flyer configuration #2 which has been previously analyzed with a RANS -to - FEAP coupling for the steady state case. The results given in this Section extends this RANS solution for the unsteady case. The RANS and FEAP meshes are shown in Fig. 6.7 thru 6.9. The total volume mesh contains about 2.95M tetrahedral elements. The CSD mesh (FEAP) surface mesh contains about 6.3K quad elements. This particular set up does not include internal structure.

Flow Conditions

Table 6.2 gives the flow conditions for this case. The flow was assumed to be laminar everywhere at Mach = 5 at 30 km altitude and angle of attack of 5 degrees. The wall temperature was held fixed at 300K. All units are in MKS. Table 6.3 presents a "run description" for this case. It is shown to document the exact run parameters**.**

Table 6.2 Flow Conditions (kg / meter/ sec)

density	**velocity**	**pressure**	**temperature**	**viscosity**	**Cp**
6.8E-4	***1491. , 130.51, 0.0***	***2.65E+4***	***261 K***	***1.75E-5***	***1006.***
Length	**Mach**	**Pr**	**Re**	**Tw**	α
20 m	***6.0***	***0.71***	***6.25E+4***	***300***	***5 deg***

Table 6.3 Run Parameters Description

```
nelem=     2957461
  npoin=      522132
  nboun=       53887

 icinf dens u-velo v-velo w-velo pres/mach
     1   5.8e-4   1491.  130.51  0.0 43.3
  gamma  pramb      nvisc
  1.400  2.6497E+04     1
  muinf  tiinf   tsude   pr    cp    tbous
 1.75e-5   0.     0.    .71    1005.  300.
   dtmin   dtmax ntime
 0.16e-3 0.16e-3  1000

 LU-SGS implicit solver
 HLLC solver, vanAlbada limiter
 100 pseudo-timesteps per implicit timestep
   5 LU-SGS passes per pseudo-timesteps
 local Courant-nr: 5.10

 5 eigenmodes from FEAP
```

Unsteady Solution - RANS to FEAP

The following are unsteady steady state solution profiles for the above configuration and flow conditions.

Figure 6.7 shows the RANS surface mesh in the top portion and the surface and symmetry plane meshes in Figs. 6.8 and 6.9. The pressure loads were passed to FEAP where the stresses and displacements were computed. In addition, the viscous loads, as computed in the RANS solution, were also passed to FEAP.

The unsteady motion is driven by the structural eigenmodes. In this case, we used 5 such eigenmodes. Figure 6.10 is a plot of 4 displacement and velocity eigenmodes vs. time.

Mach number contours at 4 time points in the unsteady solution are given in Fig. 6.11. The pressures at the same time points are shown in Fig. 6.12.

The mesh velocity plot is very instructive. We set up the case so that the wing would deflect more than other components for display purposes. Figure 6.13 shows mesh velocities for the full configuration at the same 4 time points.

Close up plots of the wing portion is shown in Fig. 6.14. Note the relatively large deflections.

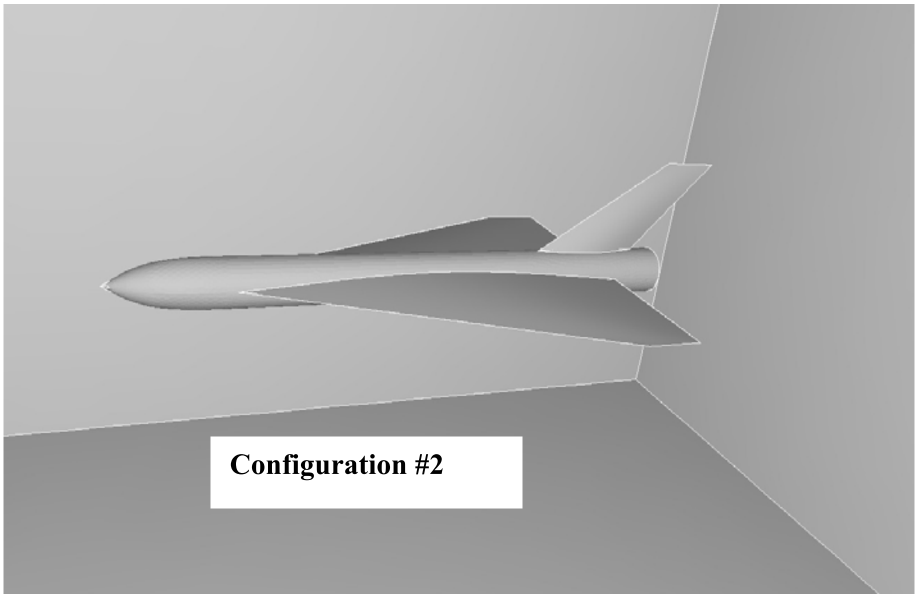
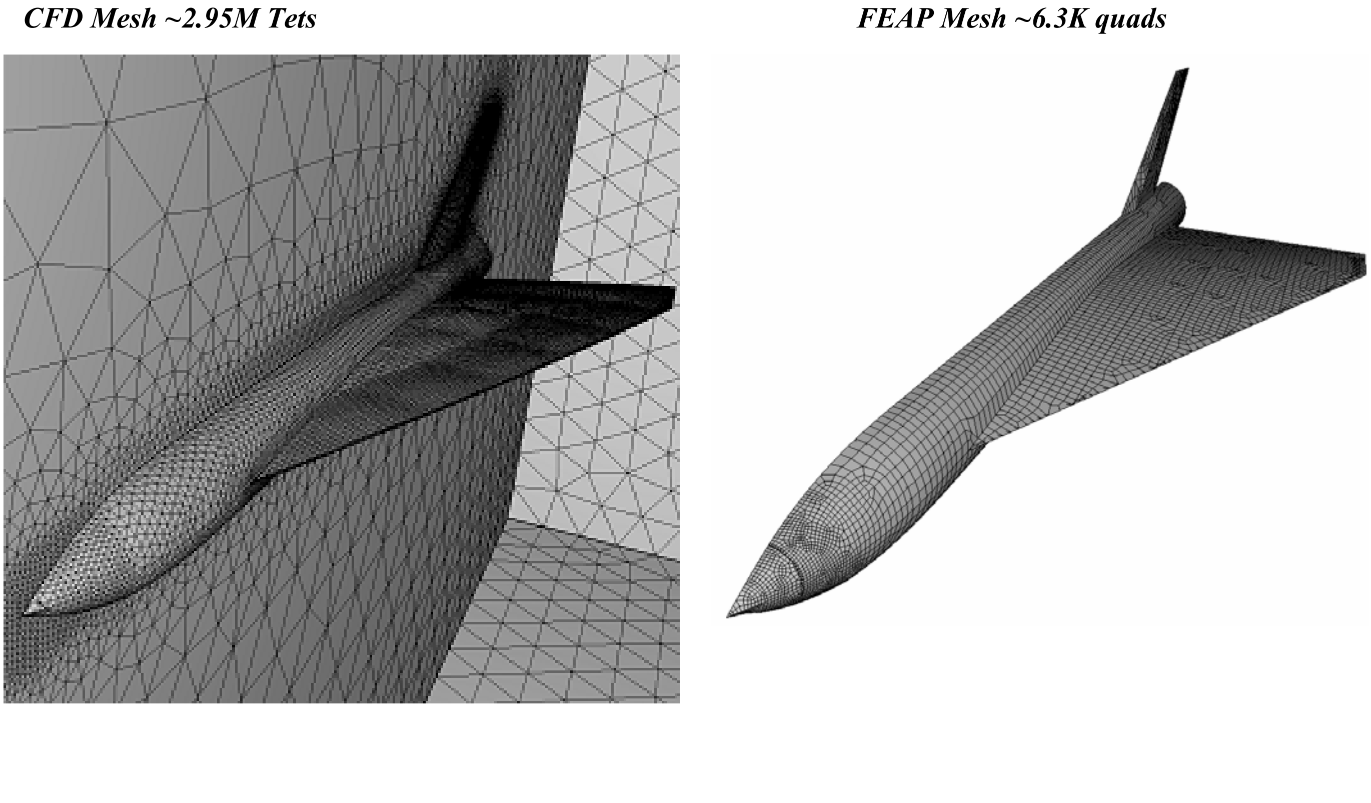


Fig. 6.6 Configuration #2 for RANS Coupling to FEAP


CFD Mesh ~2.95M Tets

FEAP Mesh ~6.3K quads

Fig. 6.7 RANS Symmetry Plane Mesh and FEAP Surface Mesh

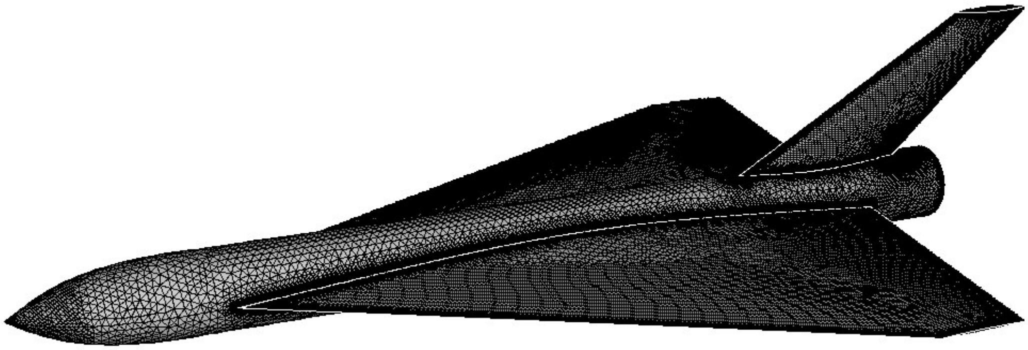


Fig. 6.8 RANS Surface Mesh

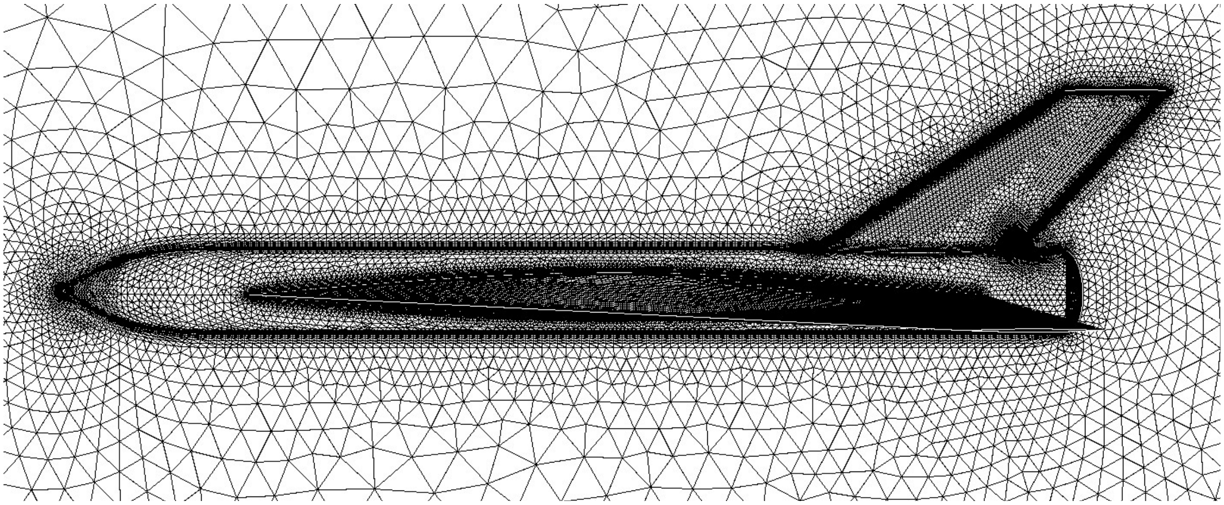
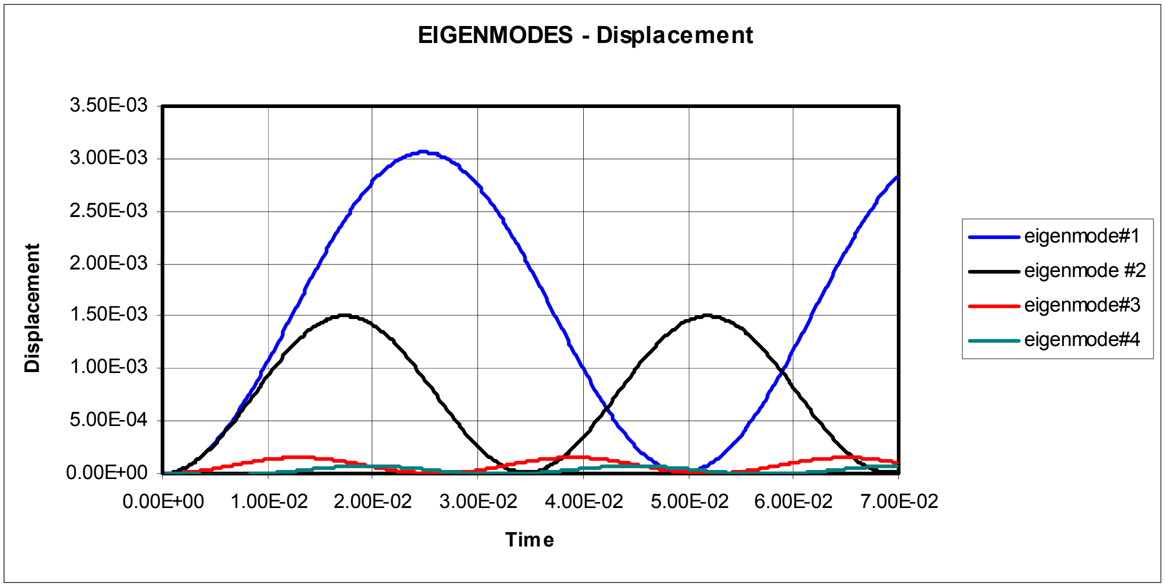


Fig. 6.9 CFD Symmetry Plane and Surface Mesh



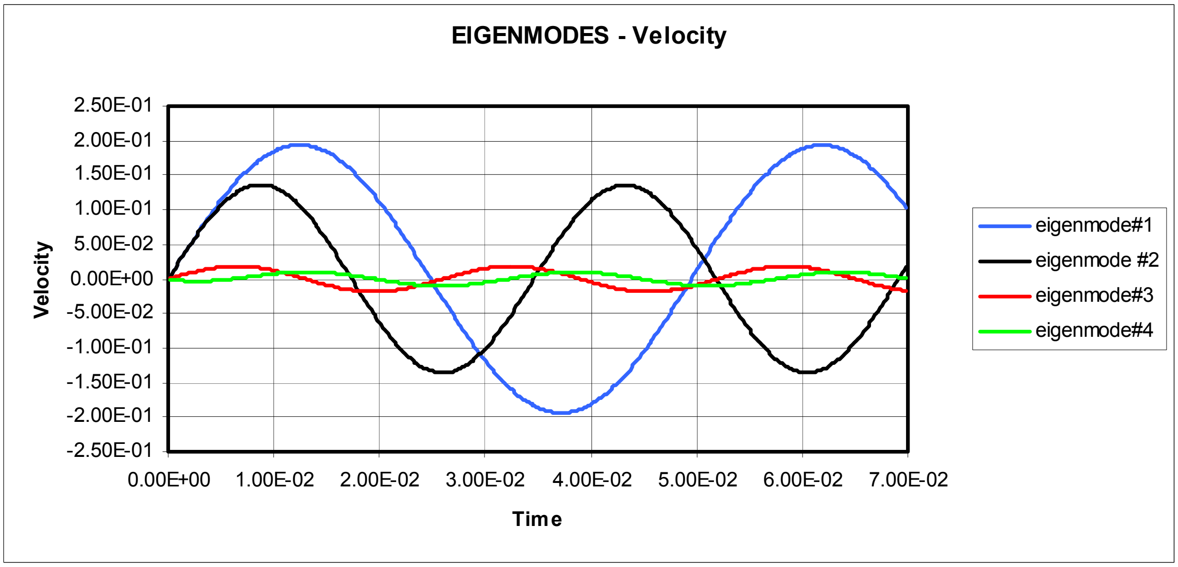


Fig 6.10 Four Eigenmodes for Generic Vehicle – Displacement and Velocity

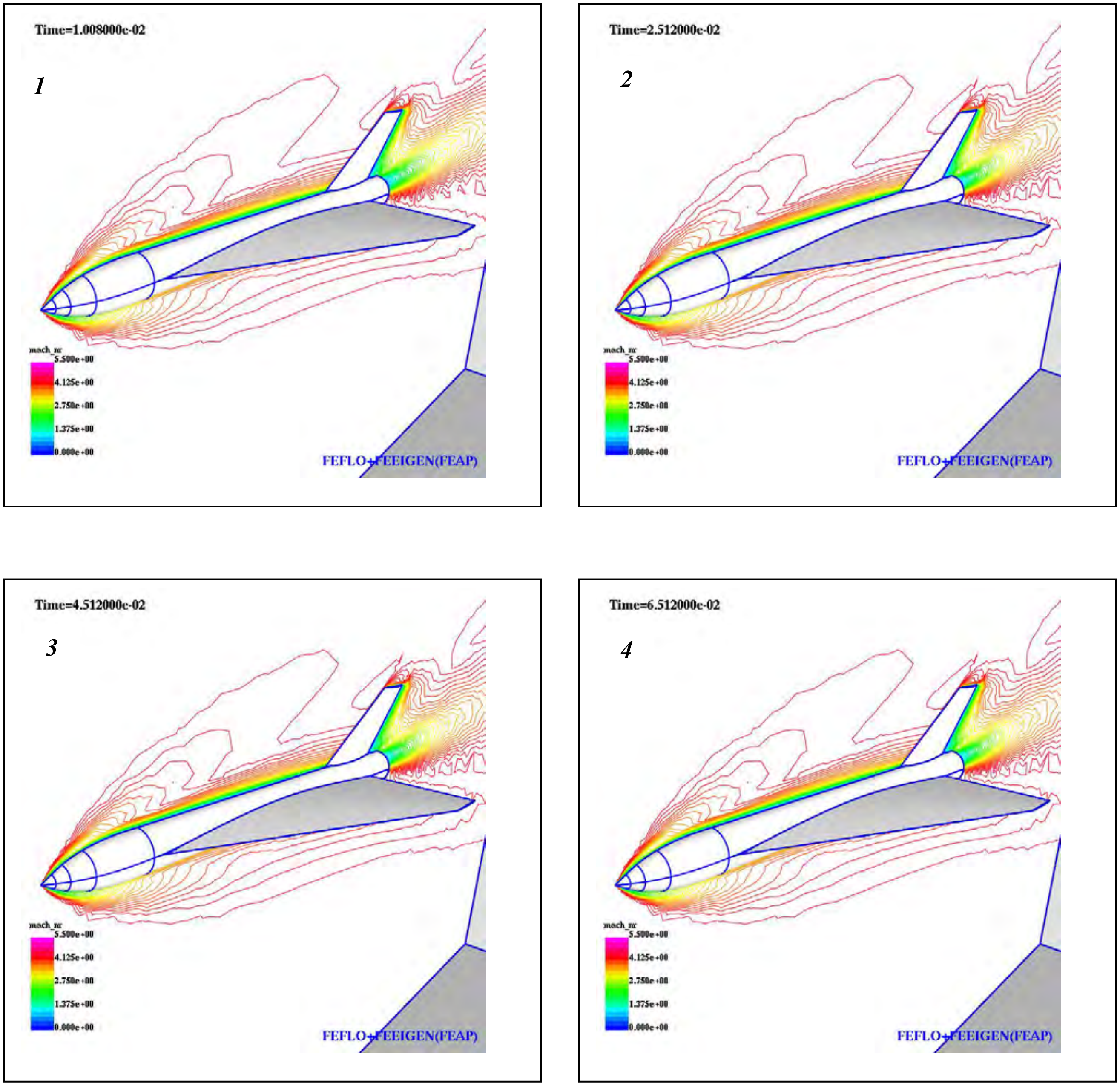


Fig. 6.11 Mach Contours at 4 Time Points

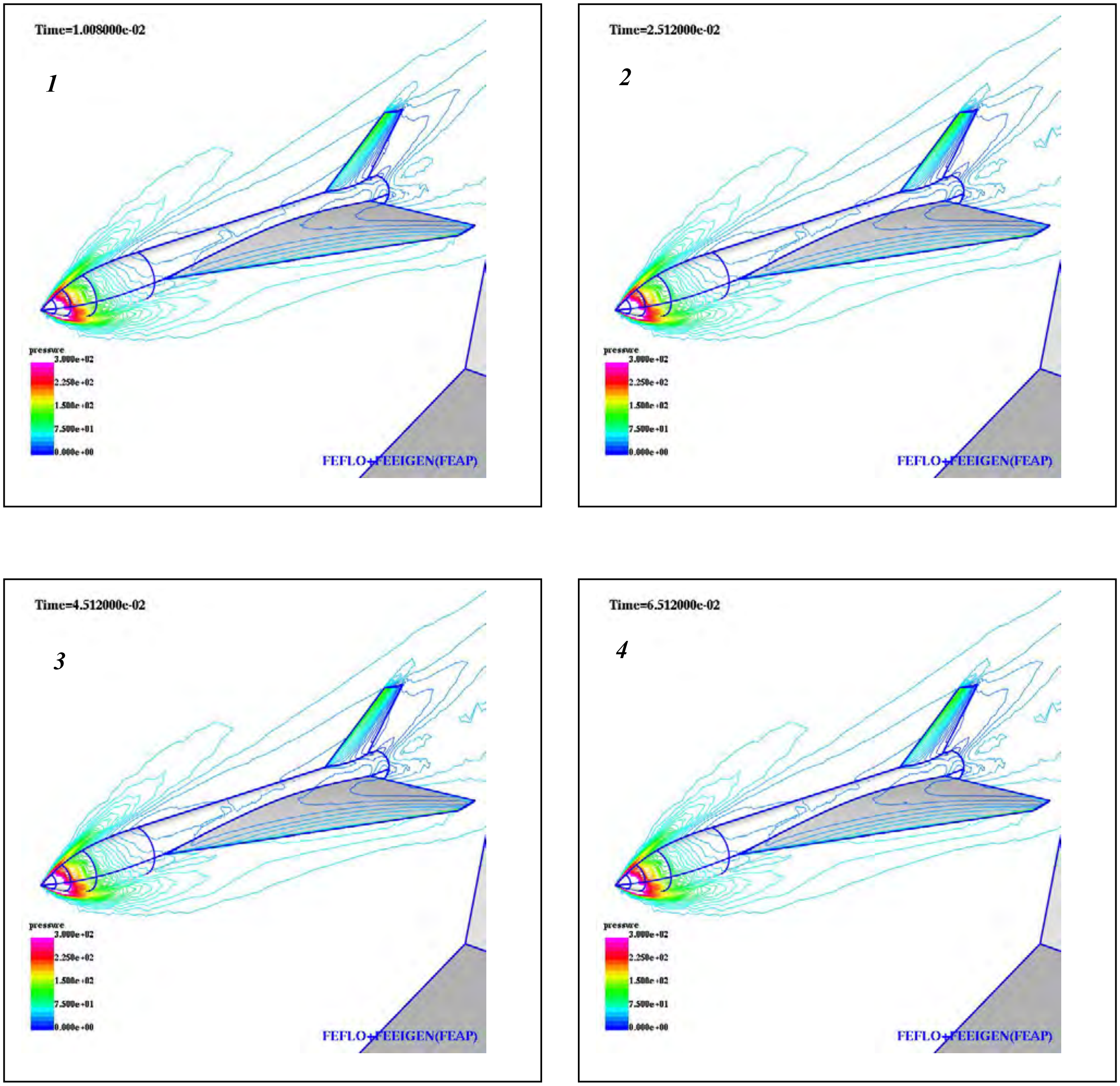


Fig. 6.12 Pressure Contours at 4 Time Points

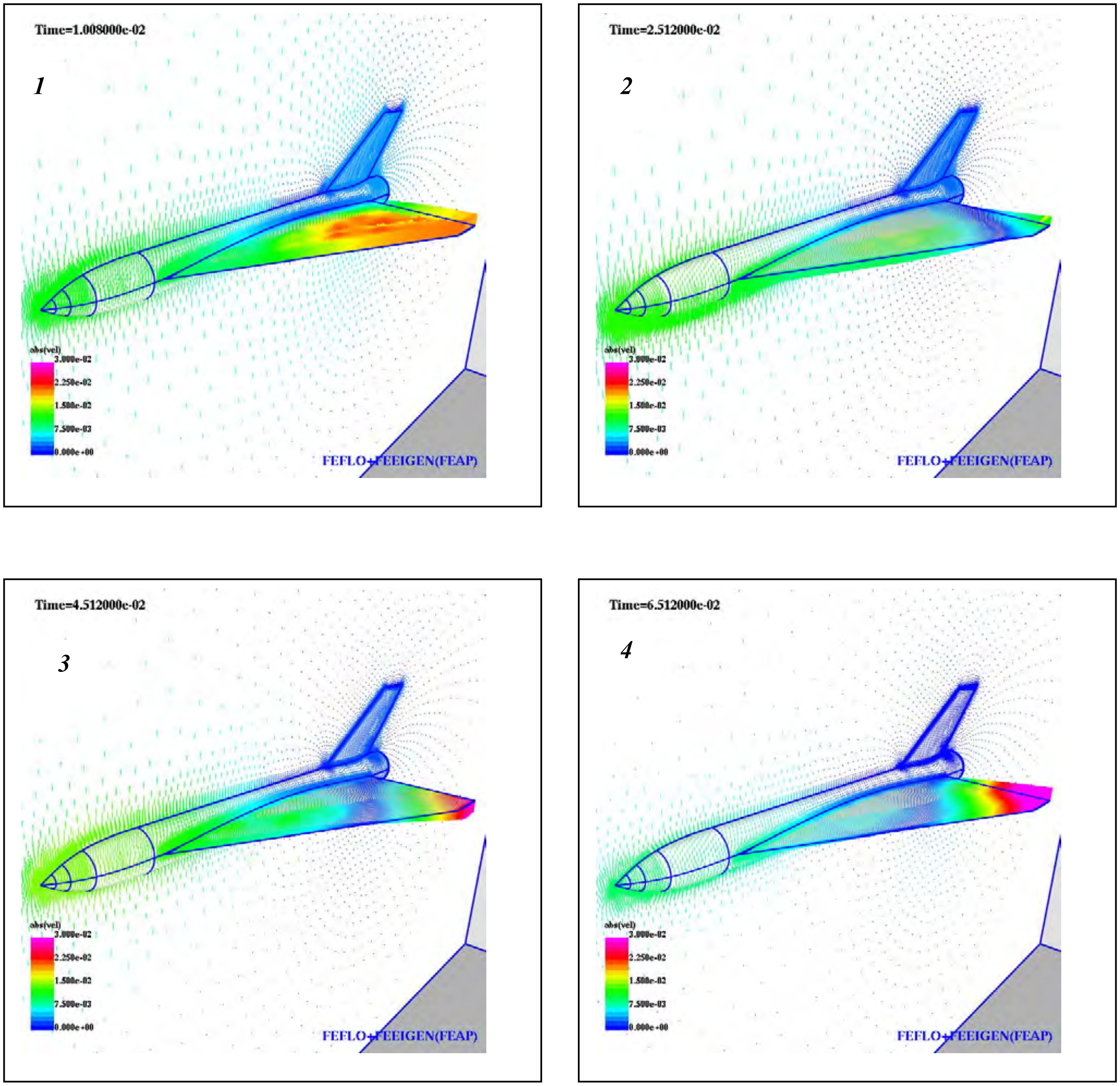


Fig. 6.13 Mesh Velocities at 4 Time Points

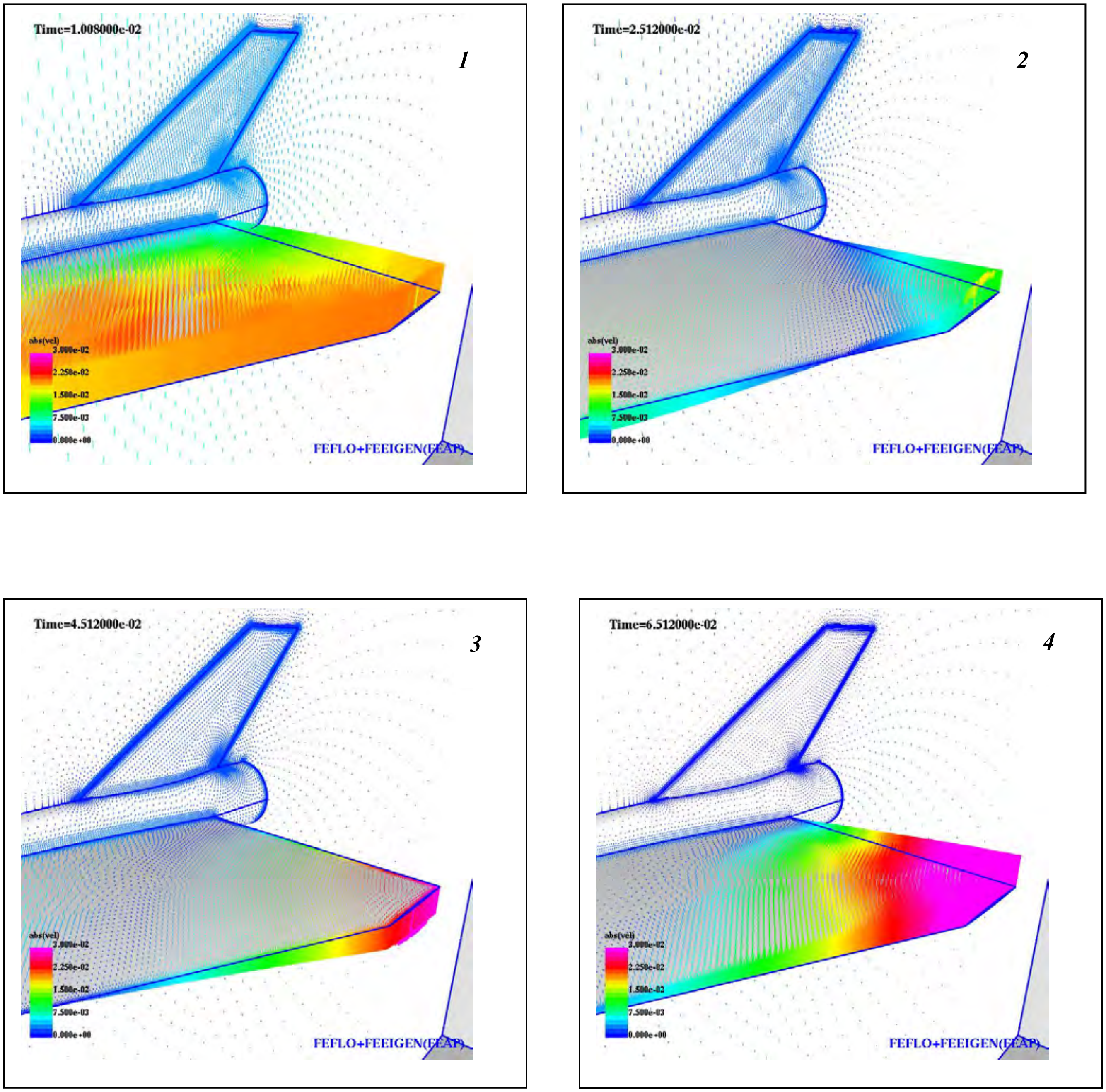


Fig. 6.14 Mesh Velocities – Close Up at 4 Time Points

7.0 PROPULSION FLOW PATH INTEGRATION

7.1 Background

The ResearchSouth contract [1] will integrate the three-dimensional CFD code FEFLO and the combustion cycle analysis code SRGULL [2] code. Software interface routines were developed to automatically pass the external flow field data to provide a three dimensional upstream flow solution at the inlet face. Work in this area has been previously conducted by Robertson and Spradley [3]. The current contract task will make use of this previous work whenever possible. Methods, subroutines, and test cases will be used for efficiency and to validate the current solutions. The work presented here gives our approach for conducting the task including a step by step procedure. A test case from this previous work is shown for illustration and for comparison to the final solution in the current contract. Porting of a beta version of SRGULL [4] to the LINUX operating system is then discussed.

7.2 Method of Approach

The problem addressed herein is defined in Fig. 7.2-1. An airbreathing vehicle configuration is defined and subjected to high-speed flow. The inviscid Euler CFD code FEFLO is used to compute the flow over the external surface with capture of the shock wave structures. The inviscid flow field is then automatically interfaced to the integral boundary layer code HUD to compute viscous effects including the boundary layer on the forebody surface. The plane of flow at the inlet face is extracted and interpolated onto a mesh for the SRGULL code to use as its upstream boundary condition. The reacting flow inside the engine is then analyzed with the SRGULL code. The exit flow is likewise interpolated onto the FEFLO mesh for computing the three –dimensional tail region.

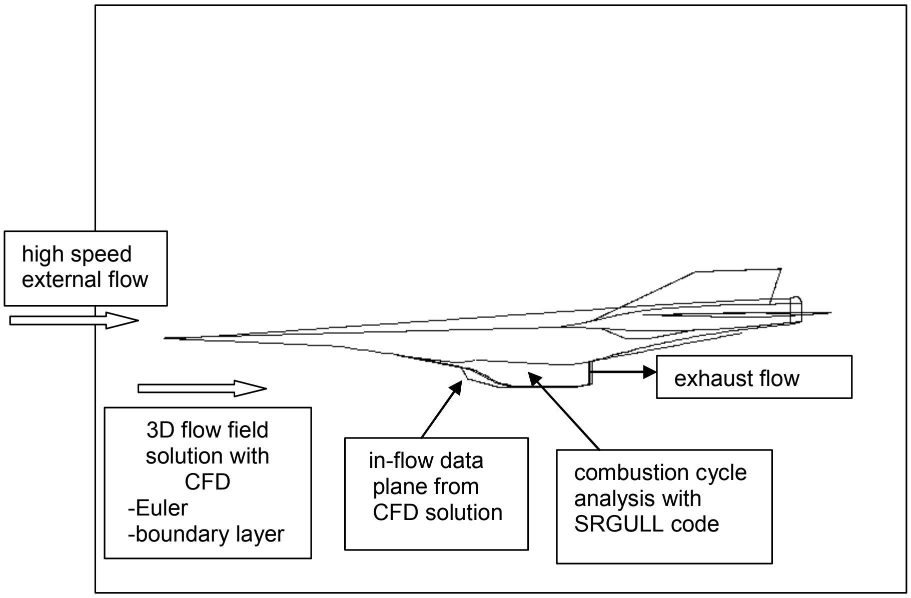

high speed external flow
exhaust flow
3D flow field solution with CFD
-Euler
-boundary layer
in-flow data plane from CFD solution
combustion cycle analysis with SRGULL code

Fig. 7.2-1 Method of Approach

Procedure Summary

1. The first step consists of a 3D Euler (inviscid) computational fluid dynamics analysis of the front end of the model, including the forebody and inlet. This step uses the FEFLO code.

2. This is followed by computation of viscous flow properties (heat transfer and shear) over the forebody and inlet using the HUD boundary layer code.

3. An interface module prepares input files with the 3D flowfield data in the proper format and converted to SRGULL variables

4. The SRGULL code is executed, using the 3D flowfield data for the model front end, to obtain the flow field in the combustor including the exit plane flow conditions.

5. A FEFLO computation of the 3D Euler flow for the tail of the vehicle in then made, including the engine exhaust. The combustor exit plane data computed in the SRGULL run is input to the tail flow model.

6. The HUD code then uses the tail inviscid flow and computes viscous flow properties over the nozzle.

Code Interface

A software interface was developed to provide the communication between the software tools. This software interface automatically passes the FEFLO / HUD flow field data to the SRGULL code to use as its initial plane of data. The effects of three-dimensional shock losses and an accurate three-dimensional mass capture are thus taken into account.

The FEFLO / HUD data consists of density, velocity, energy, shear and heating rate values assigned to the FEFLO mesh points and loaded onto a body surface file. The mesh elements are triangular face elements that cover the body surface. The body geometry data, including point coordinates and element connectivities are also loaded onto the file. These point data are then interpolated to the SRGULL surface elements that contain the FEFLO mesh points. The FEFLO mesh can be either finer or coarser than the SRGULL mesh so that each SRGULL element may be expected to contain multiple FEFLO mesh points. This is illustrated in Fig. 7.2-2. The process of interpolating the FEFLO data to the SRGULL element must be as conservative as possible within the numerical limits imposed by mesh density. The interpolation code determines an average SRGULL element data value f from

$$\bar{f} = \frac{\sum_i f_i a_i}{\sum_i a_i}$$

where f_i and a_i are FEFLO point data values and point areas, respectively, for all FEFLO data points contained within the SRGULL element. The area a_i assigned to the point i is one-third the total area of all FEFLO elements connected to the point i. The resulting average element pressures and temperatures are loaded into SRGULL.

	SRGULL	**FEFLO**
Points	**o**	**o**
Elements	━━━	───

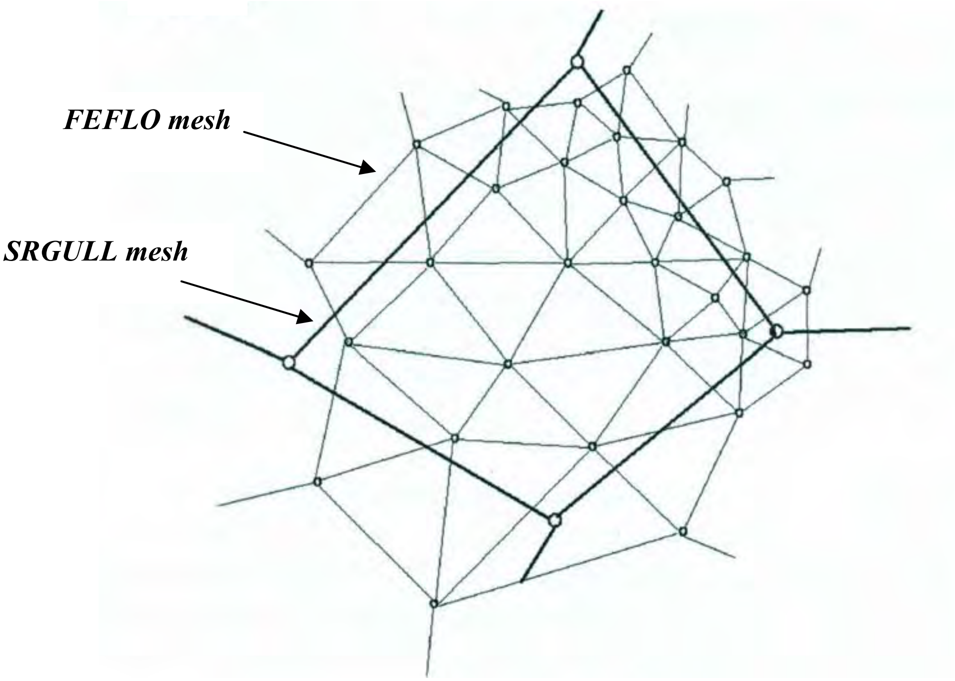


Fig. 7.2-2 Interpolation Between FEFLO and SRGULL Meshes

The FEFLO points are located within SRGULL elements by applying a series of tests. These tests determine: (1) if the point projected on the plane of the element is contained within its bounds (within a tolerance), (2) if the direction of the normals of the FEFLO elements and SRGULL elements are consistent, (3) the projected distance to the SRGULL element plane, and (4) the minimum distance of the FEFLO point to the points on the SRGULL element. The SRGULL element is selected by optimizing a combination of the above determinations. This selection process effectively selects the proper element, even with complicated geometries where multiple elements may satisfy one or more of the above determinations.

If the FEFLO mesh is sparse in the neighborhood of a SRGULL element such that no FEFLO data points are contained within the element, an alternate method is used for finding the average data value on the element. First, point values are found at each point of the SRGULL element by finding the FEFLO element containing the SRGULL element and interpolating between the three points on the FEFLO element. Next, after having found point values at the points on the SRGULL element, an average element value is found by averaging the point values on the element.

7.3 Generic Vehicle Test Case

The software is demonstrated for a generic hypersonic flyer at Mach 15 and 100,000 ft (30.48 km) altitude. A 3D view of the generic plane is shown in Fig. 7.3-1. The model is 10 ft (3.048 m.) long from front to back and 2 ft (0.6096 m.) wide.

Inviscid Forebody Flow

The computational domain for the flowfield analysis of the forebody of the vehicle model is limited to the cowl inlet up to the inlet throat and the external region forward of the cowl leading edge. The model and flowfield are bisymmetric, so the computational domain is limited to a half space bounded on one side by the symmetry plane. The boundary of the computational domain is shown in Fig. 7.3-2. The computational mesh is shown over the bounding surfaces in Fig. 7.3-3 and over the symmetry plane in Fig. 7.3-4. The mesh is refined for high resolution in the region of the bow shock on the under side of the model and inside the inlet. A close-up view of the mesh inside the inlet is shown in Fig. 7.3-5.

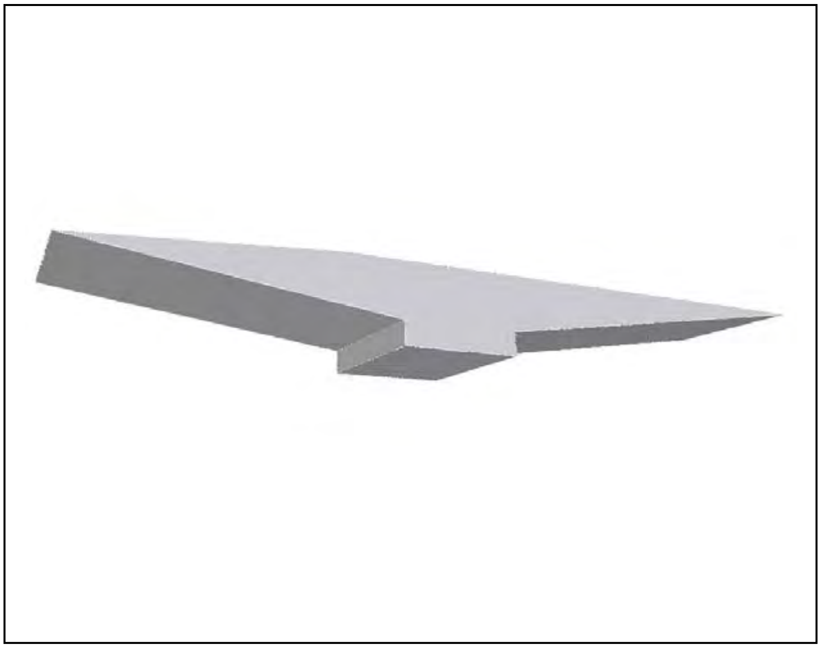


Fig. 7.3-1 Generic Flyer Model Test Case – Computational Model

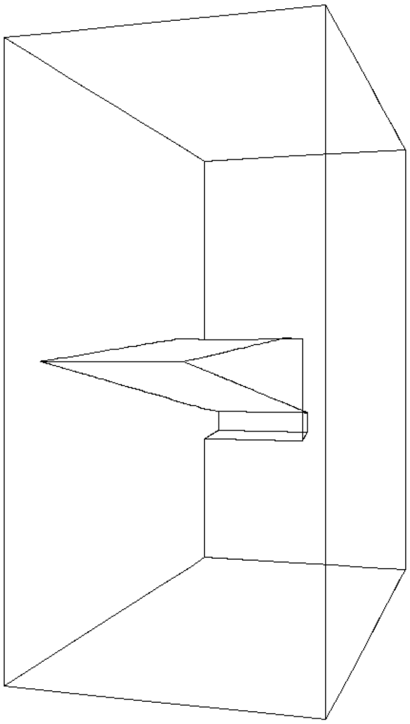


Fig. 7.3-2 CFD Domain Boundaries for Flyer Model

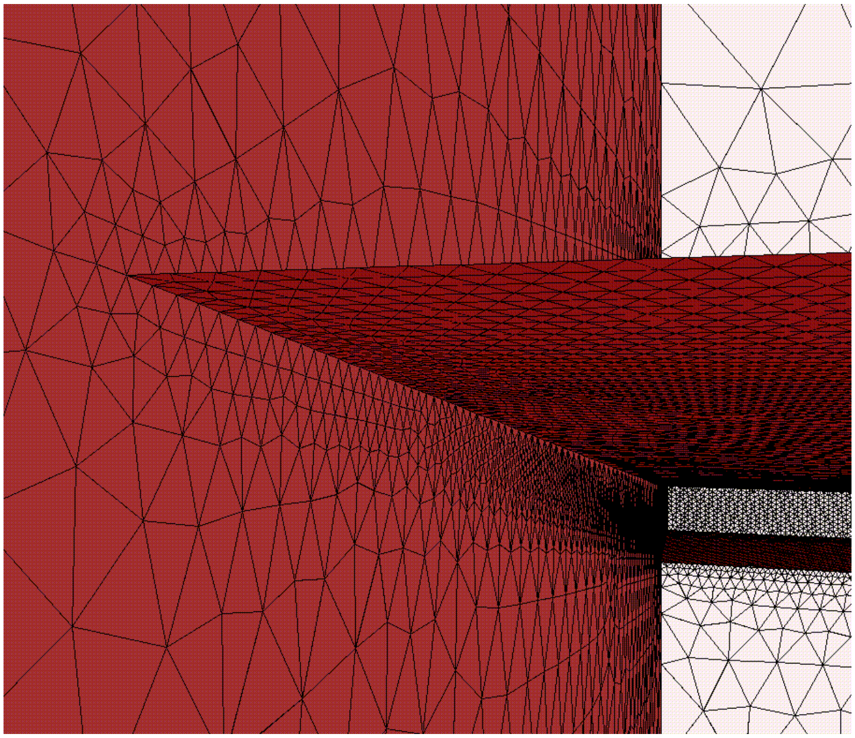


Fig. 7.3-3 Computational Mesh over Domain Boundaries for Model Forebody

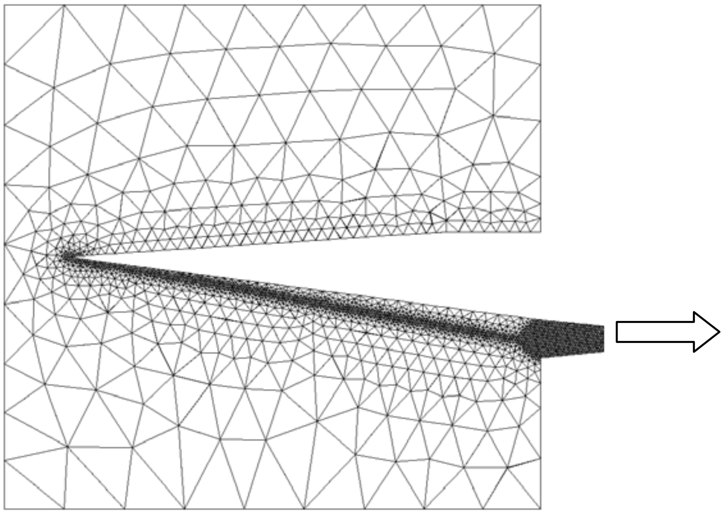


Fig. 7.3-4 Computational Mesh over Symmetry Plane

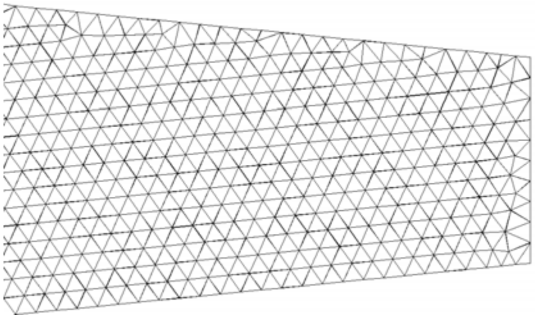


Fig. 7.3-5 Computational Mesh for Inlet

The steady state flowfield structure over the forebody and inlet is shown in the distribution of pressures over the symmetry plane shown in Fig. 7.3-6. The bow shock is captured by the cowl inlet and impinges on the cowl interior surface upstream of the inlet throat. The flowfield structure inside the cowl inlet is shown in the distribution of density in Fig. 7.3-7. The bow shock is shown to impinge and then reflect into the inlet throat.

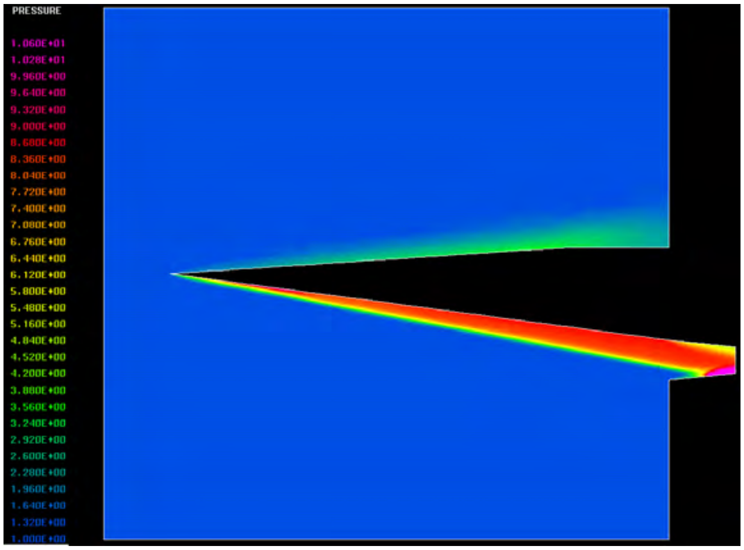


Fig. 7.3-6 Pressure Distribution over Symmetry Plane Forebody

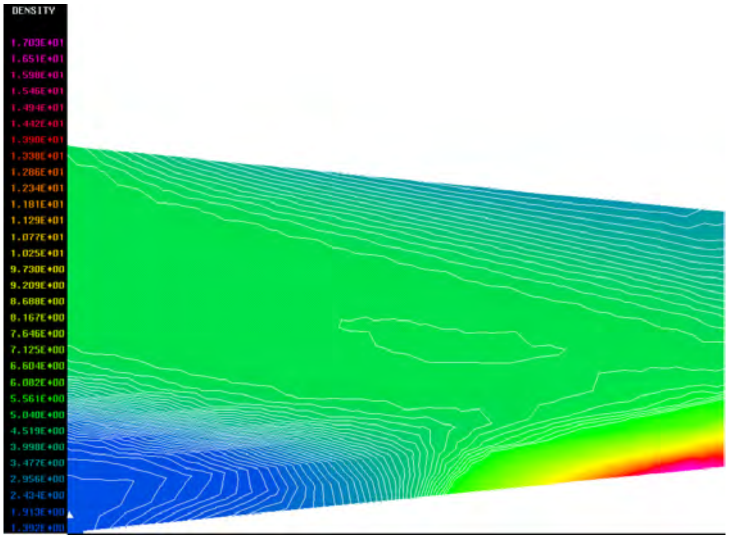


Fig. 7.3-7 Contour Plot of Density Inside Inlet

Forebody Viscous Parameters

The data files from the inviscid forebody flow are used to draw flow lines for use by the boundary layer code HUD to compute viscous parameters, shear and heating rate. Following the drawing of flow lines and the creation of a flow line data file, the solver proceeds with the viscous flow analyses along the flow lines using the HUD methodology [2].

The 3D heat flux and shear results are shown compared with corresponding 2D results in Fig. 7.3-8. Near the leading edge, the 3D results lag the 2D results due to limited resolution in the 3D computational mesh compared to the 2D solution. The accuracy of results near the leading edge and sharp corners is limited primarily by the fineness of the computational mesh in these regions. The geometry of this example model and the number of mesh points in the 3D solution were selected to enable fast run times and quick turn-around for developing and demonstrating the software.

These viscous parameters, along with Euler mass and momentum flux, and surface forces and moments, are converted to SRGULL variables and output to a file for hand - off to SRGULL for the combustion analysis.

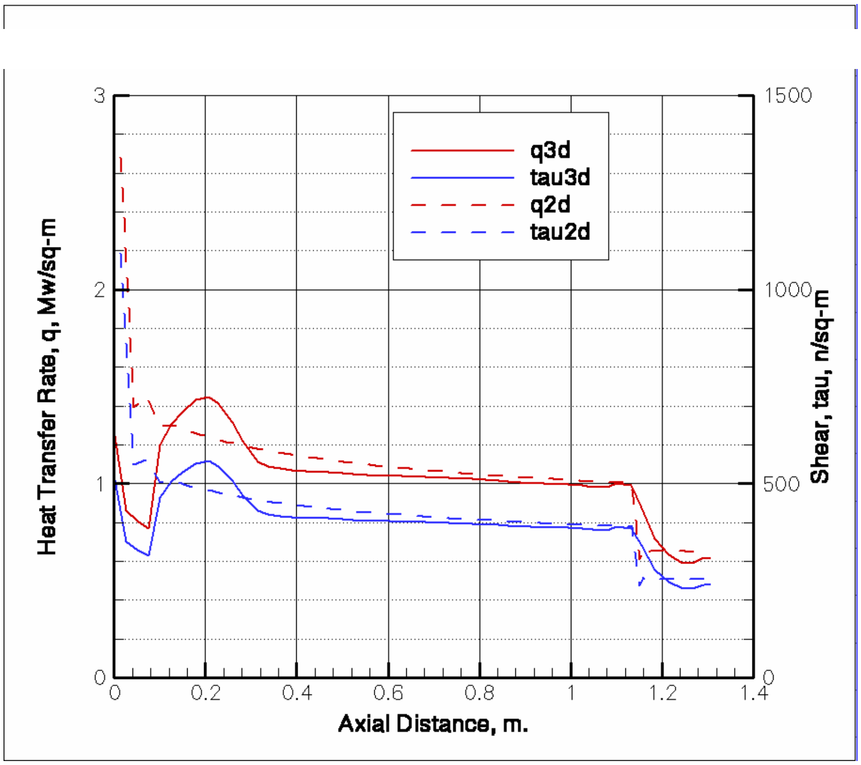


Fig. 7.3-8 Heat Flux and Shear along Flowline on symmetry Plane on Forebody From Leading Edge to Inlet Throat

SRGULL Computation

The SRGULL code uses the in-flow file from the previous step. In addition, the standard chemistry and reaction input data files are required [3]. The output of SRGULL is converted to a file format for FEFLO. These are then set as exit boundary conditions for running the tail region and nozzle sections of the problem.

Flowfield Analysis for Tail Region

The vehicle tail region was included in the overall 3D flowfield analysis by developing a computational mesh for the complete vehicle model, including the tail region with internal nozzle. For the tail analysis, the internal cowl inlet was not included since that portion of the flowfield would not affect the exterior flow about the tail The combustor exit plane conditions were included as boundary conditions for flow out of the combustor exit plane from the SRGULL file. The computational mesh over the symmetry plane for the tail region flowfield analysis is shown in Fig. 7.3-9.

The computed flowfield for the nozzle and tail is shown in the density contour plot in Fig. 7.3-10a and the Mach number contour / velocity vector plots in Fig. 7.3-10b. These plots show the well-developed nozzle flow corresponding to combustor exit plane conditions as well as the interaction of the nozzle and freestream flows in the exterior flow region.

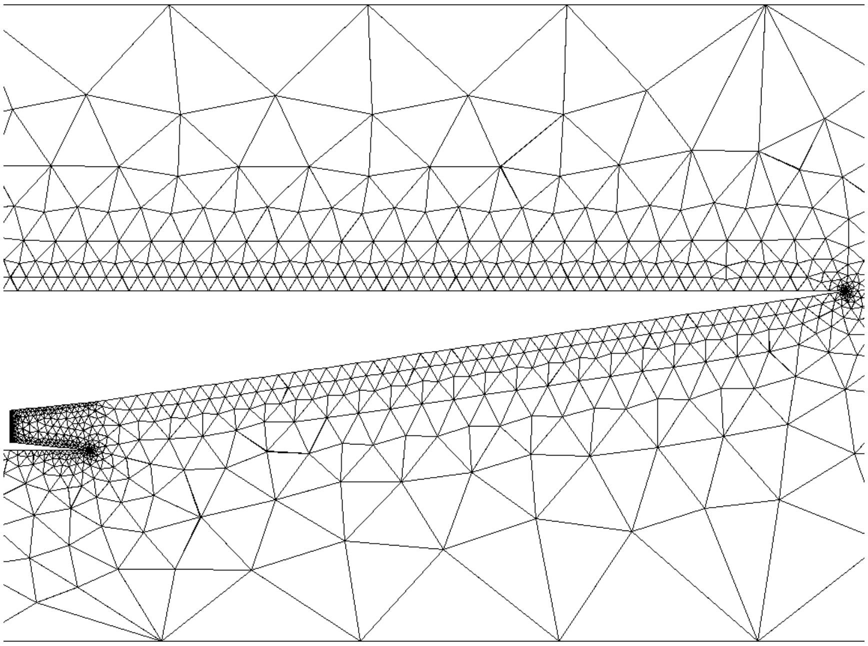


Fig. 7.3-9 Computational Mesh over Symmetry Plane for Vehicle Tail Region

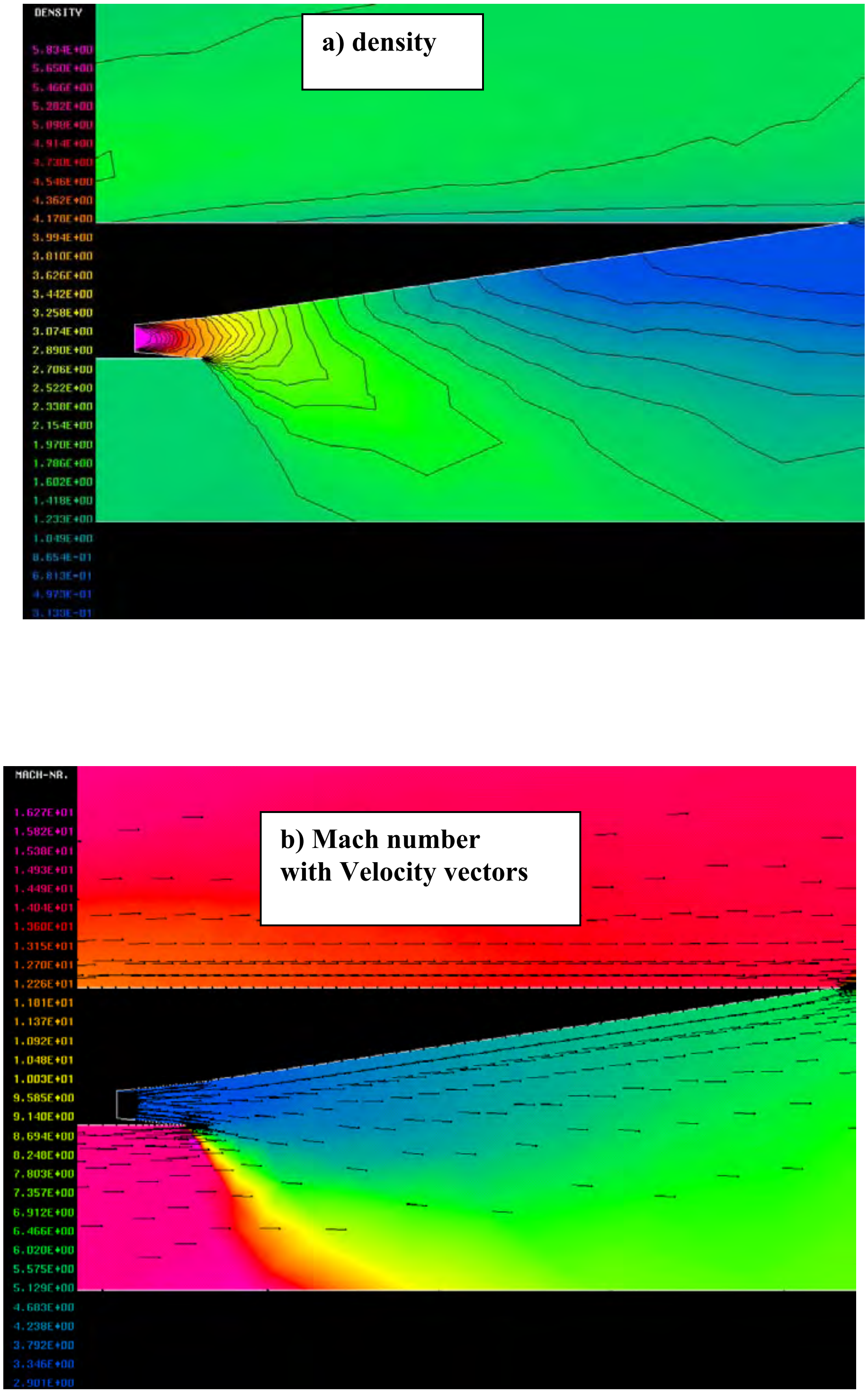


Fig. 7.3-10 Flowfield Over Symmetry Plane for Nozzle and Vehicle Tail Region

Tip to Tail Solution

The flow field for the complete model is made by assembling the parts. Figure 7.3-11 shows the symmetry plane view of the solution for density and pressure for the complete model. The surface pressure from both a front and rear view are given in Fig. 7.3-12. Mach contours at the rear symmetry plane are plotted in Fig. 7.3-13 and a rear look at the surface pressure is given in Fig. 7.3-14. This shows the engine exit plane.

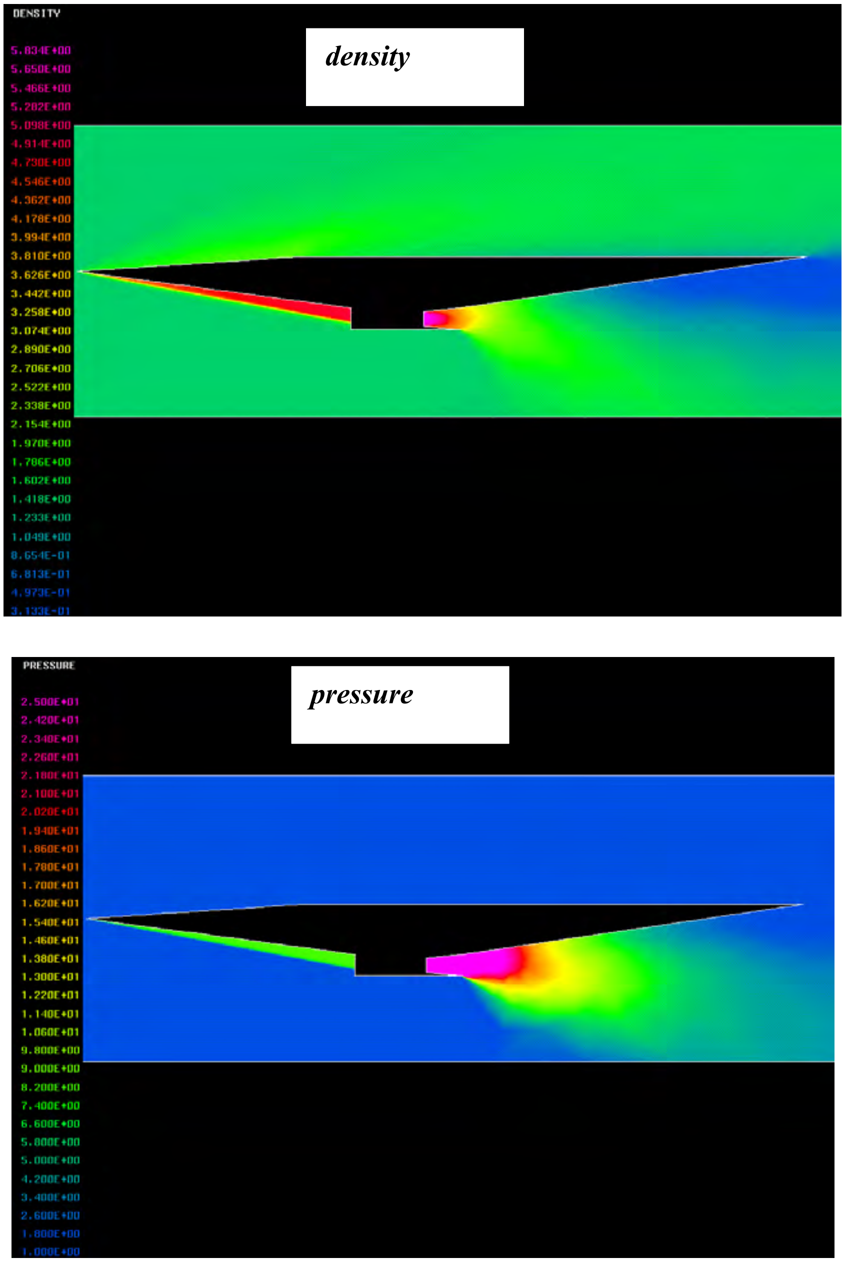
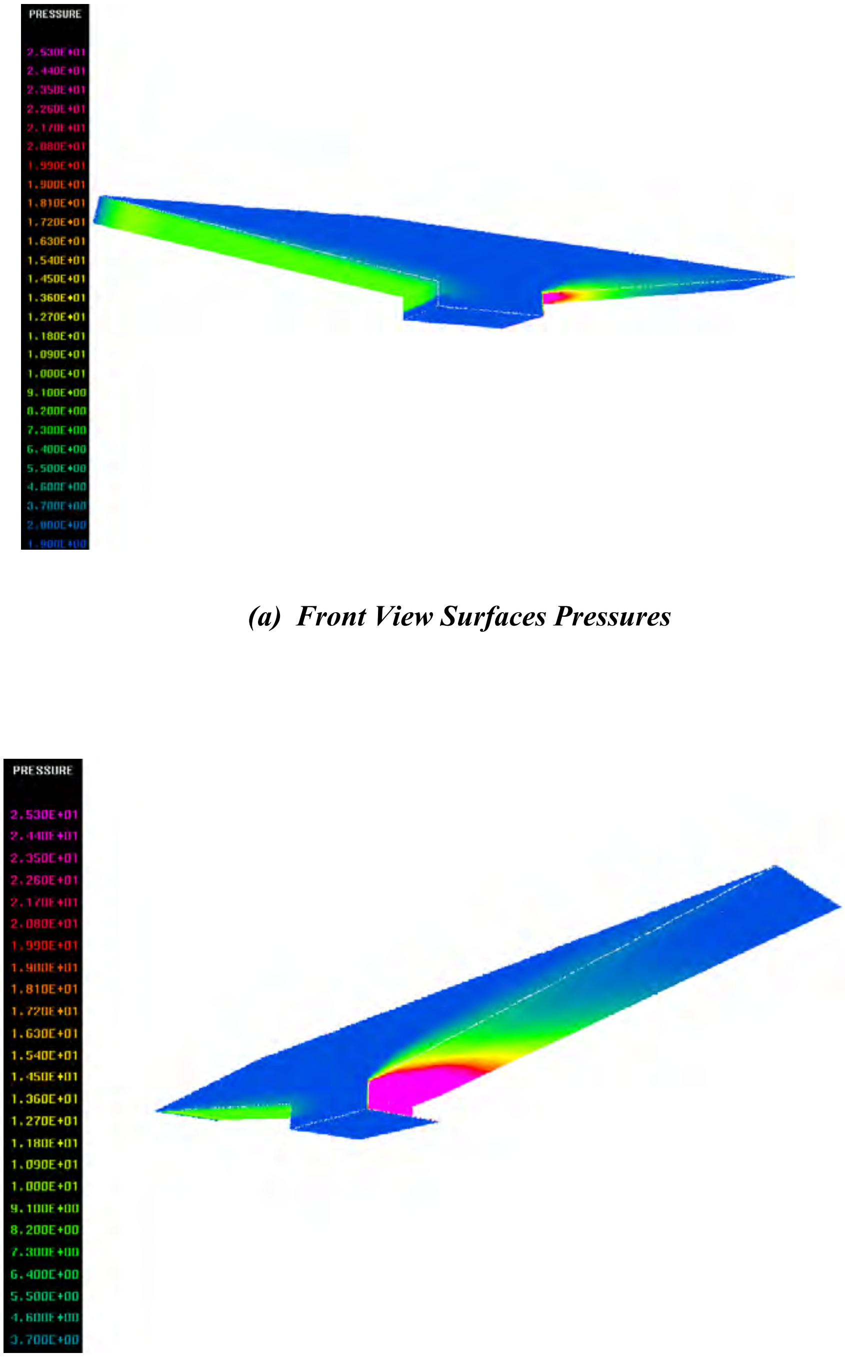


Fig. 7.3-11 Complete Vehicle Solution, Density and Pressure Contours, Symmetry Plane



(a) Front View Surfaces Pressures

(b) Rear View Surface Pressures

Fig. 7.3-12 Surface Pressures for Two Views

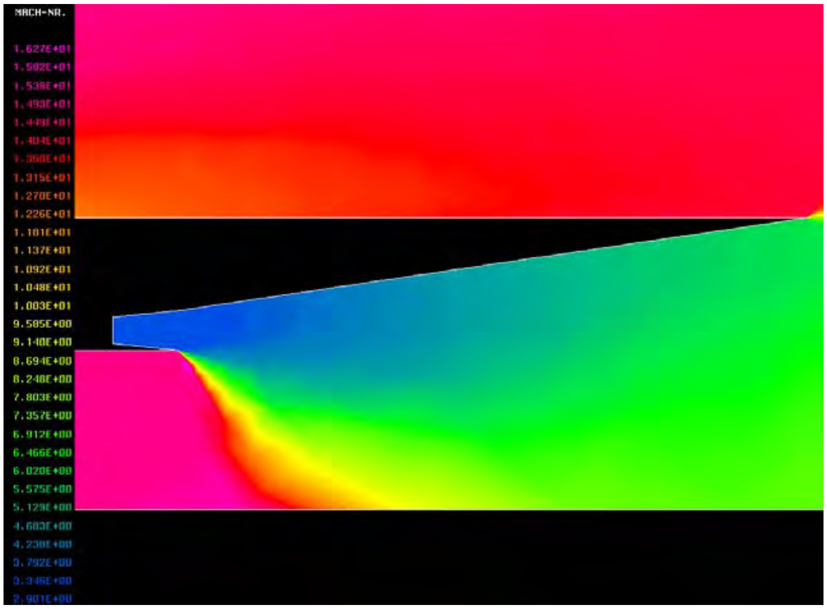


Fig. 7.3-13 Mach Number – Close up Rear Symmetry Plane View

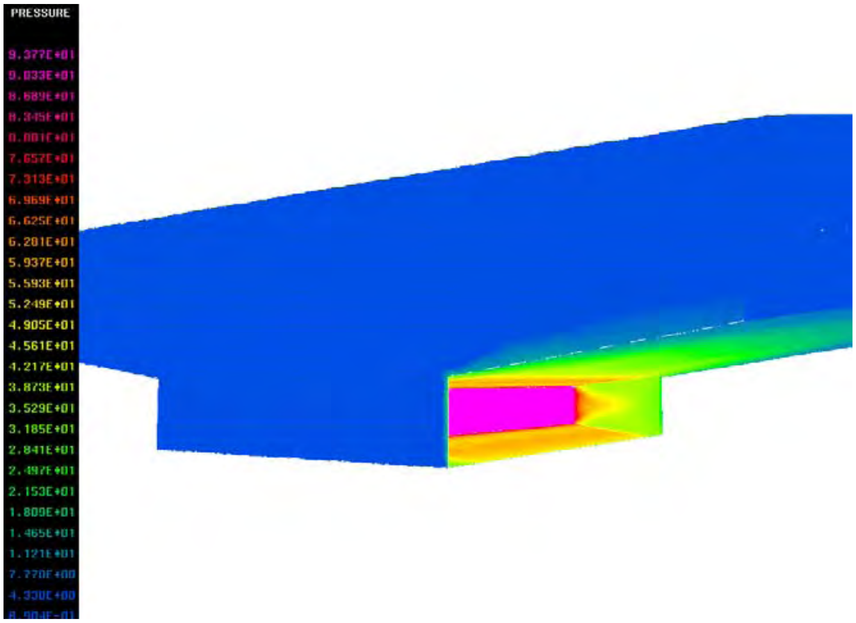


Fig. 7.3-14 Surface Pressure – Close up Rear View

Heading Rate and Shear

The heat transfer rate and shear from tip to tail is plotted in Fig. 7.3-15. The heating rate is on the left side with its scale, and the shear on the right side with its scale, all plotted versus axial distance from the nose.

The plot shows the variation in heat flux and shear along flow lines on the symmetry plane from the forebody leading edge through combustor to tail trailing edge. Each region is identified, ie forebody, inlet, combustor and nozzles.

A comparison is made between the 2D and 3D solutions for this simple model. We see the agreement is good in all regions showing the relative 2D nature of this simplified flyer model.

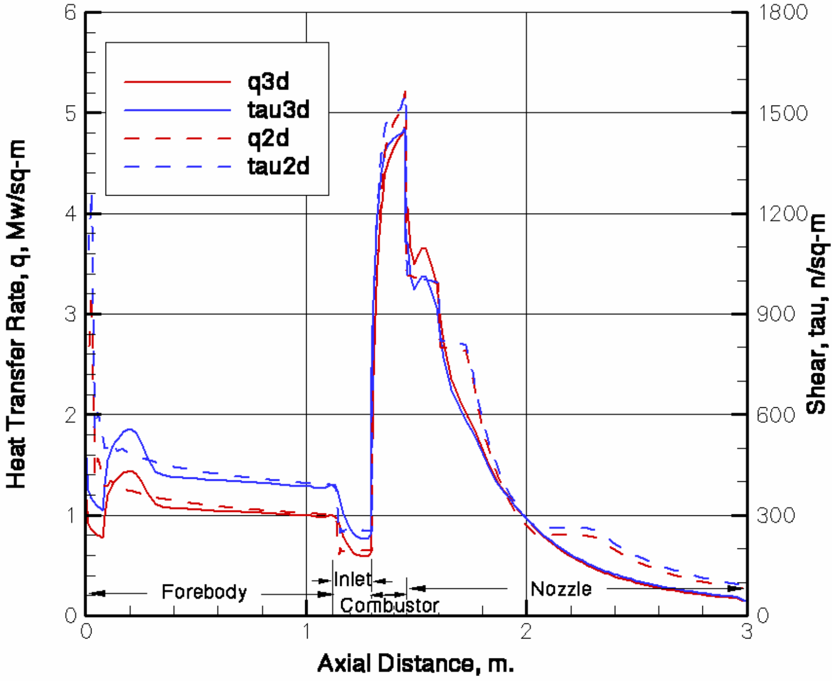


Fig. 7.3-15 Tip to Tail solution – Heating Rate and Shear vs. Axial Distance from Nose

7.4 Porting SRGULL to LINUX

In the current contract reported herein [1], ResearchSouth is to implement this propulsion flow path prediction capability to provide the thrust vector for calculating vehicle moments for establishing trim position or quasi-trim position. This capability will also apply in quantifying objective functions in optimizing the overall highly-coupled vehicle designs. This capability will consist of using the FEFLO / HUD flow predictions over the forebody and to the inlet face/entrance where the existing SRGULL inlet/isolator/combustor methodology is to be implemented and applied. The flow calculations are then to be continued with FEFLO / HUD from the combustor exit and continued through the internal and external nozzle. The current contract task will make use of this previous work whenever possible. Methods, subroutines, and test cases will be used for efficiency and to validate the current solutions.

One difference that we found in performing the current task is the incompatibility of the SRGULL UNIX version with the modern PC's running the LINUX operating system. To resolve these differences, we obtained a recent of SRGULL [4] from NASA Langley, which is a “beta” version. Therefore, we had to port this LINUX version to our PC’s at ResearchSouth.

Summary of Test Cases

The LINUX version of SRGULL from NASA Langley was compiled on two different machines at ResearchSouth. It was compiled on a LINUX machine running a CENTOS 5.3 64 bit operating system using two different compilers (Intel and gfortran). The second machine is a Laptop running Mac OSX 10.6. Only gfortran compiler was used on the Laptop. Compilation was done with no compiler options except default and then with specific compiler options. SCRAMJZ test case was used as the baseline and results compared to Langley results where possible [2].

The SRGULL compiled with all options using gfortran, but some gave warnings. Compiling with Intel in the Centos 6.3 was very sensitive to the compiler options. Using options which allows “loose” coding, we get a compilation, but with warnings. When we “tighten” up options to look for bad coding, the compilation failed. However, even with a successful compilation, some test cases from the SRGULL manual [1] did not execute properly. Table 7.1 shows some examples of these results.

Table 7.1 Example Cases Summary

Case	Comments
LACEAUG	Terminated by user after 10 minutes wall time.
PREHEAT	Clean run. Tip to Tail plots made.
NORMALSH	Did NOT run due to input file error.
RAMJETZ	Did NOT run due to input file error.
COVENR	Clean run. Tip to Tail plots made.
SCRAMJZ	Clean run. Tip to Tail plots made.

Validation

The SCRAMJZ case, for which we have a successful run, is shown here and compared to the published NASA solution.

The flow domain for the SCRAMJZ test case is shown in Fig. 7.4-1. The flow parameters are given in [1]. Contour plots from the ResearchSouth solution are shown in Figs. 7.4-2 and 7.4-3. Pressure and Mach Number are given, including an enlarged view of the Mach contours.

Plots of tip-to-tail pressure and Mach number are given in Figs. 7.4-4 and 7.4-5. These compare the solutions computed with the beta version of SRGULL code with the solution delivered by NASA [1]. The agreement is excellent.

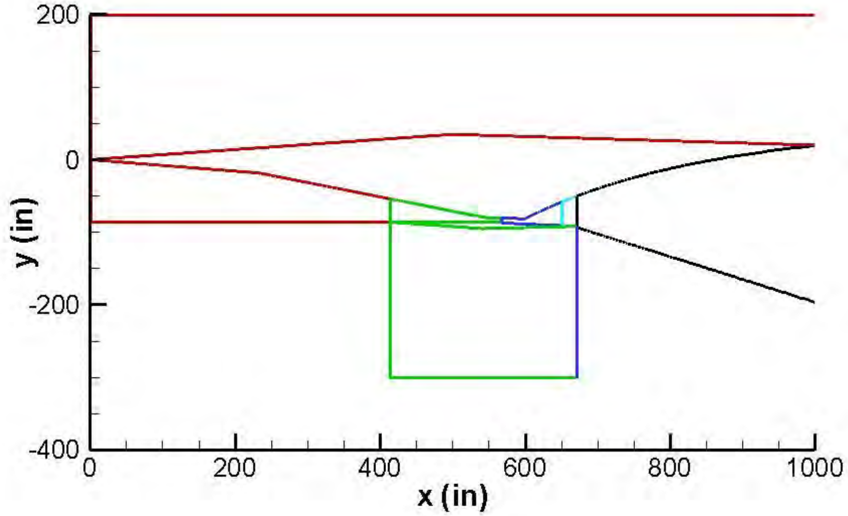


Fig. 7.4-1 Flow Zones for SCRAMJZ Case

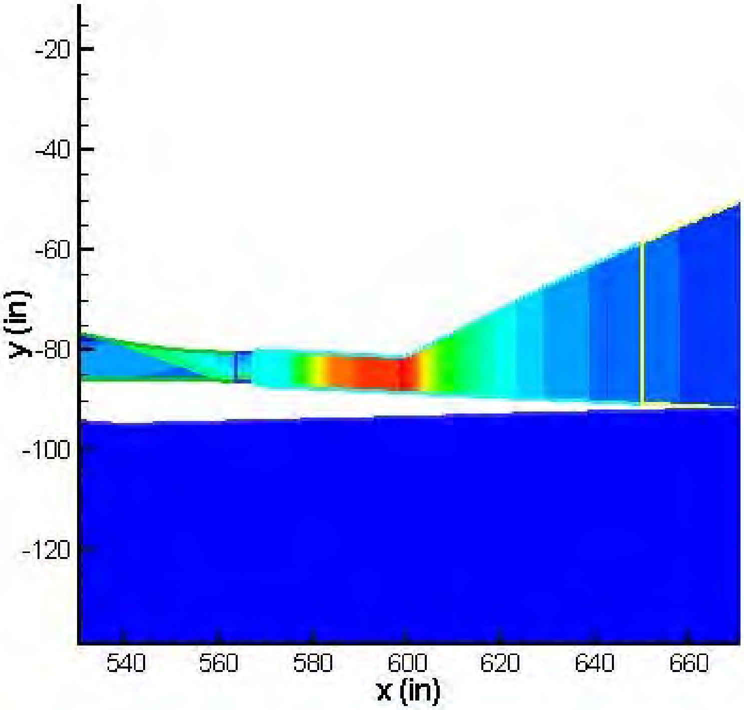
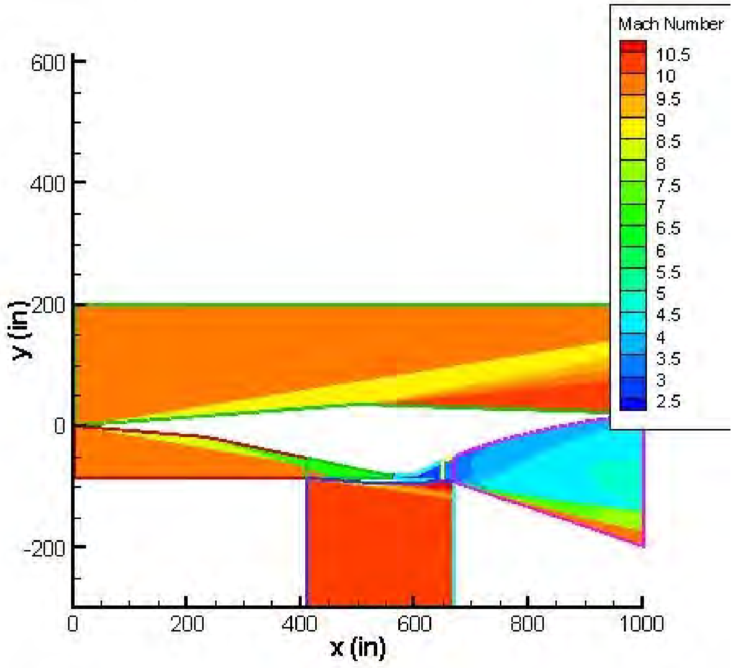


Fig. 7.4-2 ResearchSouth Solution - Pressure contour plot for SCRAMJZ Case

Full view



Enlarged view

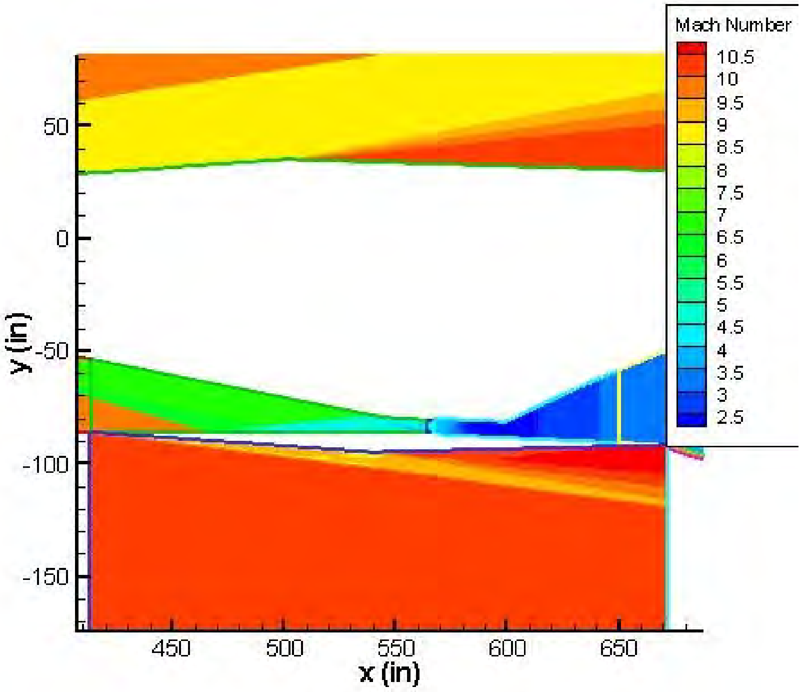
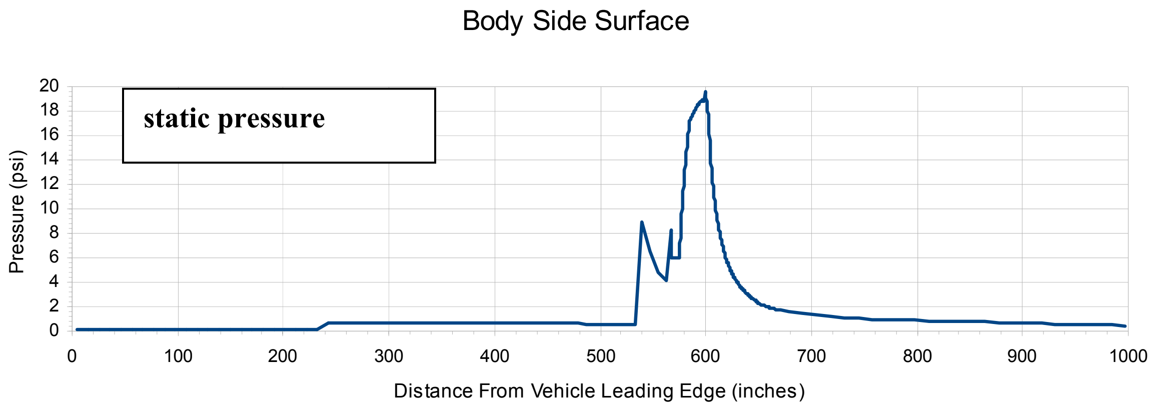
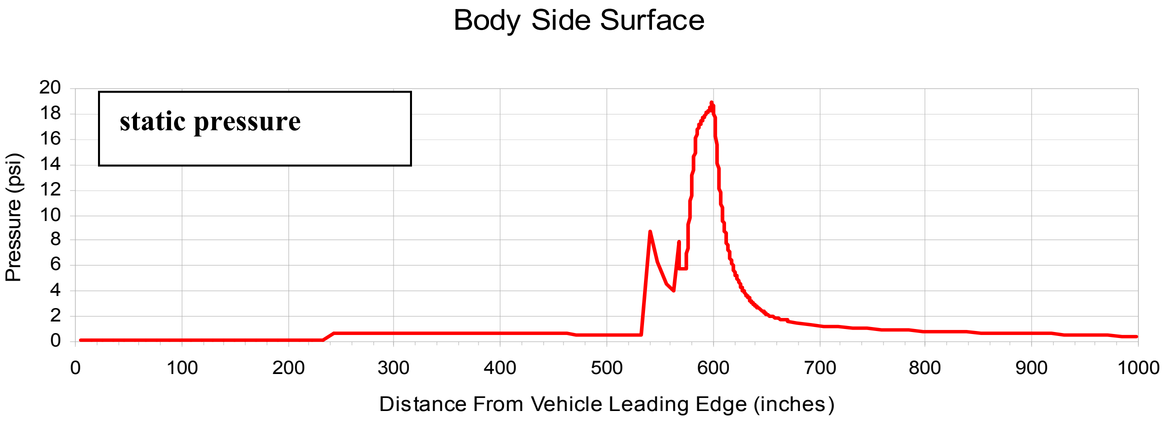


Fig. 7.4-3 Mach Number plot for SCRAMJZ case

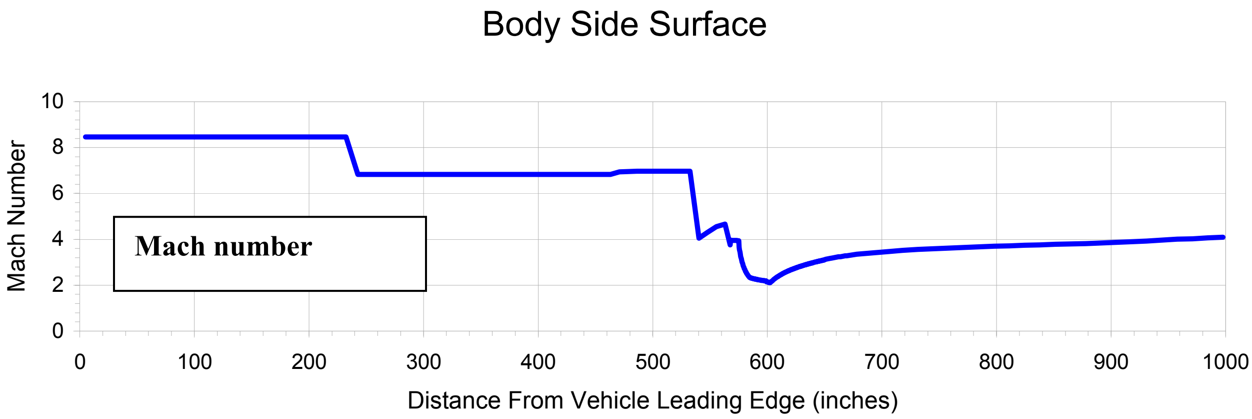


a) ResearchSouth Solution

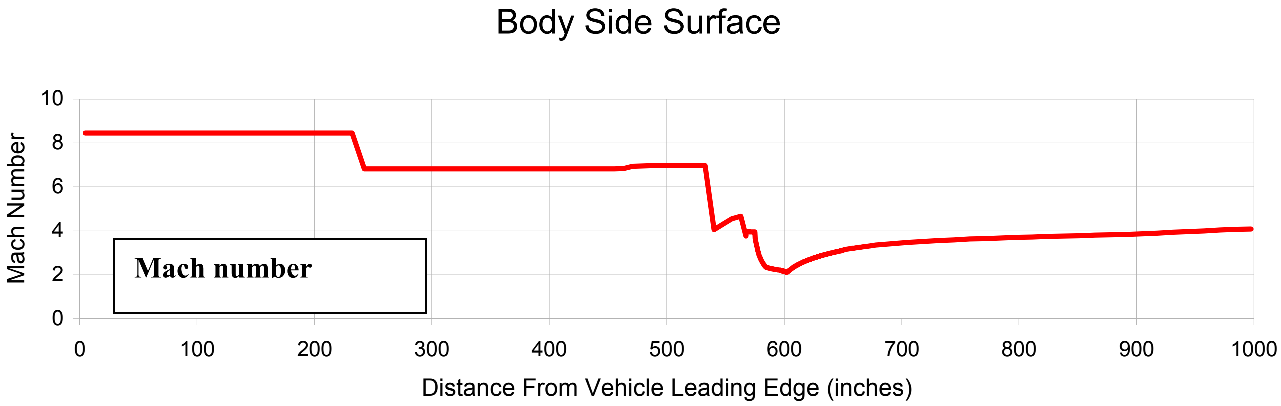


b) NASA Langley Solution - SRGULL CD

Fig. 7.4-4 Tip-To-Tail Pressure Plot for SCRAMJZ Case – Comparisons



ResearchSouth Solution



NASA Langley Solution - SRGULL CD

Fig. 7.4-5 Tip-To-Tail Mach Number Plot for SCRAMJZ Case - Comparisons

7.5 Recommendations

The SRGULL code has undergone continuous development over the past three decades. There have been many contributors and with many programming styles. In addition, SRGULL has been ported to many computational environments over the years. Today, SRGULL is very useful code, but requires an expert to properly use it. It requires multiple input files and user input errors are not uncommon. Much effort and care is required when porting the code to a new machine or compiling with a different compiler. There are many sections of SRGULL that use iterative schemes and can lead to convergence issues. Solutions on one computer can be different from solutions on another computer. SRGULL is a validated, useful state of the art code for integrated engine/airframe performance but requires care and much effort for day to day use by non experts.

As a result of our contract effort, we recommend a re-construction effort to enhance and extend the usefulness of SRGULL into the next decade. The upgrades which we recommend will increase the value of the code as an evaluation and assessment tool for hypersonic propulsion combustion concepts for current and future vehicles. We suggest a two Phase approach:

Phase I: Four tasks will be performed during Phase I. Task 1 is to create a baseline version of SRGULL source code that reflects a common coding convention based on FORTRAN 90 programming practices. Memory management and iterative schemes will be updated. Task 2 will consist of two subtasks: initial SRGULL knowledge base creation and automatic source code generator development. This knowledge base will consist of the baseline source code and sample thermodynamic and transport properties data sets. A PYTHON-based automatic source code generator will produce problem-specific categorized source code using the knowledge base library. Task 3 will consist of running a sample case from SRGULL manual and the results compared to published data. Task 4 will be to document Phase I activities and deliver an initial version of the source code, the knowledge base and the source code generator to NASA.

Phase II: The Phase II will consist of four tasks. First, an extensive SRGULL-specific XML schema and model description language will be developed and implemented. Second, the SRGULL knowledge base will be extended to include run templates, expanded thermodynamic and transport properties data sets and SRGULL documentation including training material from expert user experiences. Third, additional PYTHON-based tools will be developed to facilitate coupling of SRGULL to other codes, such as MATLAB [5], ParaView [6], CFD codes [7], sizing codes and optimization codes. Fourth, a set of extensive validation runs will be made and results compared to NASA approved results. At the end of Phase II, the complete source code and knowledge base will be delivered to NASA, documentation provided, and a training class given.

This re-constructured SRGULL code will then replace the version used in this contract effort and the integration with FEFLO can be automated more completely.

7.6 References

[1] Spradley, Lawrence, and Lohner, Rainald, " Multidisciplinary Modeling Software for Analysis, Design, and Optimization of HRRLS Vehicles, Annual Report #2, Contract NNL08AA33C, ResearchSouth, Inc, March 30, 2010.

[2] SRGULL Code, Version 2, LAR-16290-1, November 2009.
Pinckney, S.Z., "Program ZMCNEW: Second Generation Engineering Model for the Prediction of Airframe-Integrated Subsonic / Supersonic Combustion Ramjet Cycle Performance".

[3] Robertson, James, and Spradley, Lawrence, "Integration of CFD and Combustion Cycle Analysis", Final Report, Contact NAS1-00133, ResearchSouth, Inc., January 2001.

[4] Ferlemann, Shelly, "SRGULL – Beta Version for LINUX", Private Communication, 2010.

[5] http://www.mathworks.com/.

[6] http://www.paraview.org/

[7] http://www.scs.gmu.edu/~rlohner/pages/publications/fluidstructint.html

8.0 AERODYNAMIC SHAPE OPTIMIZATION

8.1 Adjoint Method

The aerodynamic and fluid dynamic design optimization methodology adopted for this research effort follows that developed earlier by Jameson et al and described in [1] for use with the Navier-Stokes equations. Lohner and co-workers [2] have adapted these methods for use with unstructured computational fluid dynamics (CFD) analysis methodology. These design optimization methods employ adjoint counterparts to the partial differential equations (PDE's) that describe the physical environment to be optimized. The partial differential equations may include inviscid or viscous fluid dynamics (Euler, Navier-Stokes) equations, magneto hydrodynamics equations, or solid mechanics equations with elastic-plastic-fracture deformation. The following expanded derivation is intended to enhance understanding of the theory underlying the methodology.

In general, a design optimization involves maximizing or minimizing some functional involving geometrical or physical variables describing the system. Examples are maximizing lift or minimizing drag on some aerodynamic body:

$$I_c = Lift = \int p \ n_y \ d\Gamma \tag{8.1}$$

Or

$$I_c = Drag = \int p \ n_x \ d\Gamma \tag{8.2}$$

The functional I_c is referred to as a cost function which is to be maximized or minimized within the bounds of certain imposed constraints which include the governing PDE's as well as other possible geometrical or physical constraints. In the examples of equations 8.1 and 8.2, the functional to be optimized is a surface integral over the aerodynamic body surface. Other cost functions could consist of volumetric integrals over portions of the spatial domain. Note that the example cost functions contain the flow variable pressure *p* along with the geometric shape of the aerodynamic body.

In general, the cost function may be represented as

$$I_c = I_c(\beta, U(\beta)) \tag{8.3a}$$

$$= \int_{\Gamma} I_c^{\Gamma}(\beta, U(\beta))\, d\Gamma + \int_{\Omega} I^{\Omega}(\beta, U(\beta))\, d\Omega \tag{8.3b}$$

where Γ and Ω refer to surface and volumetric integrals, respectively. The cost function is represented as being a function of a matrix of design variables β and the environment variables U. The environment variables U are also functions of the design variables β. The design variables usually involve parameters determining the geometric shape of surface boundaries which, in turn, determine the quantitative value of the environment variables. In the example cases, the design variables β might be the set of points describing the aerodynamic body surface or a set of equations describing segments of the surface. The environmental variable U is the matrix of flow field variables. Note that in the example cases the pressure p is the only flow field variable appearing explicitly in the cost functions. Obviously, a change in the shape of the aerodynamic body produces a change in the pressure distribution over the body.

A small change in the design variables β changes the cost function I_c by the difference

$$\delta I_c = \sum\left(\frac{\partial I_c}{\partial \beta_i}\delta\beta_i + \frac{\partial I_c}{\partial U_j}\delta U_j\right) \tag{8.4a}$$

where the summation is over the total number of design variables β and environment variables *U*. In matrix notation, this is expressed more compactly as:

$$\delta I_c = \frac{\partial I_c}{\partial \beta}\delta\beta + \frac{\partial I_c}{\partial U}\delta U \tag{8.4b}$$

The optimum design will result in $\delta I_c = 0$. Approaching the optimum design requires that, for minimization problems, $\delta I_c < 0$ until $|\ \delta I_c\ |$ becomes very small, and for maximization problems, $\delta I_c > 0$ until $|\ \delta I_c\ |$ becomes very small.

Constraining the optimization will be the governing PDE's for the physical environment and their boundary conditions.

$$R(\beta, U(\beta)) = 0 \tag{8.5}$$

Equation 8.5 may be interpreted as representing the dependence of the environment variables U on the design variables β, or $U = U(\beta)$. For the example lift or drag optimization cases, the PDE equation set, R, consists of the Euler or Navier-Stokes equations. Differencing Eq. (8.5) yields:

$$\delta R = \frac{\partial R}{\partial \beta}\delta\beta + \frac{\partial R}{\partial U}\delta U = 0 \tag{8.6}$$

In constrained optimization problems, since $\delta\ R = 0$, each equation defining a constraint can be added to the cost function I_c through the use of Lagrange multipliers. The governing equation set R consists of a number of PDE's equal to the number of environment variables U, and this equation set must be satisfied throughout the domain. At every point in the domain, we define a matrix of Lagrange multipliers ψ corresponding to the matrix of environment variables U. After dividing the domain into infinitesimally small volume elements d *Ω*, then taking the product of these Lagrange multipliers with Eq. 8.6 and summing with Eq. 8.4 over the domain, we obtain the volumetric integral:

$$\delta I = \frac{\partial I_c}{\partial \beta}\delta\beta + \frac{\partial I_c}{\partial U}\delta U + \int_{\Omega} \psi^T \left(\frac{\partial R}{\partial \beta}\delta\beta + \frac{\partial R}{\partial U}\delta U\right) d\Omega = 0 \tag{8.7}$$

Where the transpose matrix ψ^T is a row matrix of the Lagrange multipliers at each point. After re-grouping terms corresponding to $\delta\beta$ and δU, we have:

$$\delta I = \left(\frac{\partial I_c}{\partial \beta} + \int_{\Omega} \psi^T \frac{\partial R}{\partial \beta} d\Omega\right)\delta\beta + \left(\frac{\partial I_c}{\partial U} + \int_{\Omega} \psi^T \frac{\partial R}{\partial U} d\Omega\right)\delta U = 0 \tag{8.8}$$

By requiring

$$\frac{\partial I_c}{\partial U} + \int_{\Omega} \psi^T \frac{\partial R}{\partial U} d\Omega = 0 \tag{8.9}$$

the variation of the augmented cost function becomes:

$$\delta I = \left(\frac{\partial I_c}{\partial \beta} + \int_{\Omega} \psi^T \frac{\partial R}{\partial \beta} d\Omega\right)\delta\beta \tag{8.10}$$

The gradients of the cost function with respect to the design variables, therefore are:

$$G^T = \frac{\partial I}{\partial \beta} = \frac{\partial I_c}{\partial \beta} + \int_{\Omega} \psi^T \frac{\partial R}{\partial \beta} d\Omega \tag{8.11}$$

where G^T is a row matrix of gradients corresponding to the design variables *β*.

The solution of Eq. 8.9 provides a set of *ψ* values that can be used in Eq. 8.11 to compute gradients for any number of design variables.

Adjoint Formulation for Conservation Law Equations

Consider the set of conservation law equations in index form:

$$R_j = \frac{\partial F_j^{\,i}}{\partial x_i} - s_j = 0 \qquad (8.12)$$

where the flux variables F and source terms s are functions of the conserved variables U, and the x_i variables are the standard rectilinear coordinates. The Einstein summation convention for repeated indices holds here with i = 1,2,3. Then Eq. 8.9 in index form becomes:

$$\frac{\partial I_c}{\partial U_k} + \int_\Omega \psi_j \frac{\partial}{\partial U_k}\left(\frac{\partial F_j^{\,i}}{\partial x_i} - s_j\right) d\Omega = 0 \qquad (8.13)$$

Changing the order of differentiation:

$$\frac{\partial I_c}{\partial U_k} + \int_\Omega \psi_j \left(\frac{\partial}{\partial x_i}\frac{\partial F_j^{\,i}}{\partial U_k} - \frac{\partial s_j}{\partial U_k}\right) d\Omega = 0 \qquad (8.14)$$

Defining the flux Jacobians A and source Jacobians C:

$$A_{j,k}^{\;i} = \frac{\partial F_j^{\,i}}{\partial U_k}\text{, and } C_{j,k} = \frac{\partial s_j}{\partial U_k} \qquad (8.15a,b)$$

and their transpose:

$$(A_{k,j}^{\;i})^T = \frac{\partial F_j^{\,i}}{\partial U_k}\text{, and } (C_{k,j})^T = \frac{\partial s_j}{\partial U_k} \qquad (8.16a,b)$$

and re-arranging terms, Eq. 8.14 becomes:

$$\frac{\partial I_c}{\partial U_k} + \int_\Omega \left[\frac{\partial}{\partial x_i}(A_{k,j}^{\;i})^T - (C_{k,j})^T\right]\psi_k \, d\Omega = 0 \qquad (8.17)$$

which in more compact matrix notation is:

$$\frac{\partial I_c}{\partial U}+\int_{\Omega}[\frac{\partial}{\partial x_i}(A^i)^T-C^T]\psi\,d\Omega=0 \tag{8.18}$$

Here ψ is a column matrix of the Lagrange multipliers.

The derivative terms in Eq. 8.18 can be integrated by parts as follows:

$$\int_{\Omega}\frac{\partial}{\partial x_i}(A^i)^T\psi\,d\Omega=\iiint\frac{\partial}{\partial x_i}(A^i)^T\psi\,dx_i dx_{i+1}dx_{i+2}$$

$$=\iint(A^i)^T\psi\,dx_{i+1}dx_{i+2}-\iiint(A^i)^T\frac{\partial\psi}{\partial x_i}dx_i dx_{i+1}dx_{i+2}$$

$$=\int_{\Gamma}(A^i)^T n_i\psi\,d\Gamma-\int_{\Omega}(A^i)^T\frac{\partial\psi}{\partial x_i}d\Omega \tag{8.19}$$

where n_i are the components of the surface normal, and the i, i+1, and i+2 indices are rotated as follows: (1,2,3), (2,3,1), (3,1,2).

By incorporating Eq. 8.19 into Eq. 8.18, and recalling the cost function definitions from Eqs. 8.3, we have:

$$\int_{\Gamma}\frac{\partial I_c^{\Gamma}}{\partial U}d\Gamma+\int_{\Omega}\frac{\partial I_c^{\Omega}}{\partial U}d\Omega+\int_{\Gamma}(A^i)^T n_i\psi\,d\Gamma-\int_{\Omega}(A^i)^T\frac{\partial\psi}{\partial x_i}d\Omega-\int_{\Omega}C^T\psi\,d\Omega=0 \tag{8.20}$$

After separately grouping the boundary and volumetric integral terms, then setting each to zero, we have the adjoint PDE's:

$$-(A^i)^T\frac{\partial\psi}{\partial x_i}-C^T\psi=-\frac{\partial I_c^{\Omega}}{\partial U} \tag{8.21}$$

and their boundary conditions:

$$(A^i)^T n_i\psi=-\frac{\partial I_c^{\Gamma}}{\partial U} \tag{8.22}$$

The Lagrange multipliers ψ in the adjoint PDE's are the adjoint variables corresponding to the conserved variables in the governing equations (Eq. 8.12). Note that the development of the adjoint PDE's led to a sign change in the first derivative convection terms. This results in a reversal in the direction of flow in the adjoint variables from that of the environment variables.

Adjoint Navier - Stokes Equations

The Euler equations are included in the set of conservation law equations represented by Eq. 8.12 without the source terms. The Euler equations for compressible flow, along with the corresponding conserved variables *U* and flux variables *F,* are given as follows:

$$R=\frac{\partial F^i}{\partial x_i}=0 \qquad U=\begin{pmatrix}\rho\\ \rho u_j\\ \rho e\end{pmatrix}, \quad F^i=\begin{pmatrix}\rho u_i\\ \rho u_i u_j+p\delta_{ij}\\ u_i(\rho e+p)\end{pmatrix}, \quad A^i=\frac{\partial F^i}{\partial U} \quad \text{i, j = 1,2,3} \tag{8.23}$$

These equations may be linearized as:

$$R=A^i\frac{\partial U}{\partial x_i}=0 \tag{8.24}$$

with ρ, u_i, *e,* and *p* being the density, velocity components, total energy and pressure, respectively. The pressure for a polytropic gas is:

$$p=(\gamma-1)\rho(e-\tfrac{1}{2}u_j u_j) \tag{8.25}$$

The corresponding adjoint PDE's are:

$$-(A^i)^T\frac{\partial \psi}{\partial x^i}=-\frac{\partial I_c^\Omega}{\partial U} \tag{8.26}$$

The Navier-Stokes equations contain additional viscous terms involving second derivatives:

$$R_j=\frac{\partial F_j^i}{\partial x^i}-\frac{\partial F_{vj}^{\ i}}{\partial x^i}=0 \tag{8.27}$$

with the viscous flux terms given by:

$$F_v^{\ i}=\begin{pmatrix}0\\ \sigma_{ij}\delta_{ij}\\ u_j\sigma_{ij}+k\dfrac{\partial T}{\partial x_i}\end{pmatrix} \qquad \text{i, j = 1,2,3} \tag{8.28}$$

and the stress terms given by:

$$\sigma_{ij} = \mu(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) + \lambda \delta_{ij} \frac{\partial u_k}{\partial x_k} \tag{8.29}$$

with T, μ and λ being the temperature and the first and second coefficients of viscosity.

In formulating the adjoint equations corresponding to the Navier-Stokes equations, repeated integration by parts on terms involving second derivatives result in adjoint viscous terms without the sign change that affected the convection terms. Jameson, et. al [1] derived the adjoint Navier-Stokes equations based on generalized curvilinear coordinates. The following development applies their derivation to standard rectilinear coordinates with some simplifications. The resulting adjoint equations are:

$$-(A^i)^T \frac{\partial \psi}{\partial x_i} - \frac{\partial \widetilde{F}_v^i}{\partial x_i} = -\frac{\partial I^{\Omega}}{\partial U} \tag{8.30}$$

where, with some linearization, the adjoint viscous flux terms are given by:

$$\widetilde{F}_v^i = \begin{pmatrix} \frac{T}{\rho} k \delta_{ij} \frac{\partial \psi_5}{\partial x_i} \\ (\widetilde{\sigma}_{ij} + \hat{\sigma}_{ij}) \delta_{ij} - \sigma_{ij} \psi_5 \\ (\rho k / R) \frac{\partial \psi_5}{\partial x_i} \end{pmatrix} \qquad \text{i, j = 1,2,3} \tag{8.31}$$

with:

$$\widetilde{\sigma}_{ij} = \mu(\frac{\partial \psi_{i+1}}{\partial x_j} + \frac{\partial \psi_{j+1}}{\partial x_i}) + \lambda \delta_{ij} \frac{\partial \psi_{k+1}}{\partial x_k} \tag{8.32}$$

$$\hat{\sigma}_{ij} = \mu(u_i \frac{\partial \psi_5}{\partial x_j} + u_j \frac{\partial \psi_5}{\partial x_i}) + \lambda \delta_{ij} u_k \frac{\partial \psi_5}{\partial x_k} \tag{8.33}$$

An inspection of these adjoint viscous terms reveals that the positive sense of diffusion produced in the Navier-Stokes PDE's is retained in the adjoint PDE's.

Example Adjoint Solution

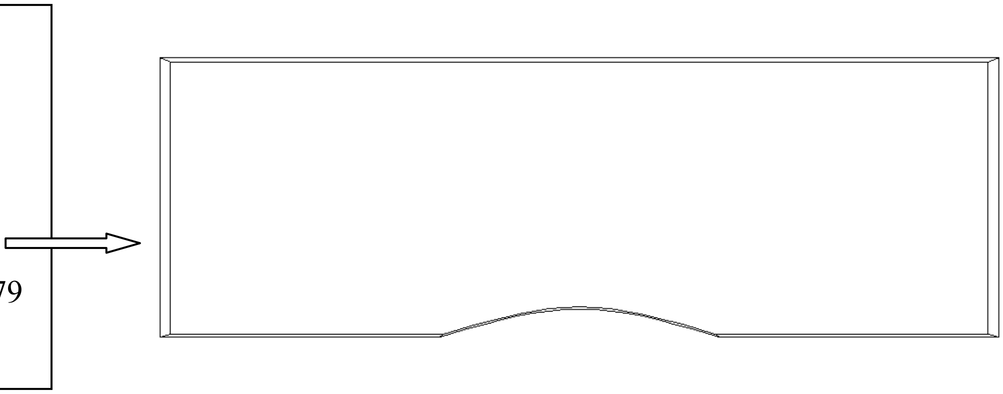
The following is a simple example that we have used to implement the solvers. We computed the supersonic flow over a simple symmetric airfoil which produces a shock in front of the body and an expansion in the back. This will test the features of the adjoint solver.

Figure 8.1-1 gives the geometry for the airfoil flow domain. The Euler flow conditions are also shown in the box. The minimum and maximum values obtained in the solution for both the Euler and the Adjoint equations are shown here for reference. The mesh used for the computation is shown in Fig. 8.1-2

The solution is shown in figures 8.1-3 thru 8.1-6. The density, absolute velocity (speed), specific energy, and velocity vectors are plotted. The Euler solution is given on the top of the figure and the corresponding Adjoint solution is shown underneath it. Comparison of the two figures shows the Adjoint solution is in fact "opposite or inverse" to the Euler solution as it should be. We are now confident that our FEFLO Adjoint code for the three-dimensional compressible Euler equations is performing properly.

Euler flow conditions

Density = 1.0
U vel = 1.0
V vel = 0.0
W vel = 0.0
P = 0.179
Mach = 2.0



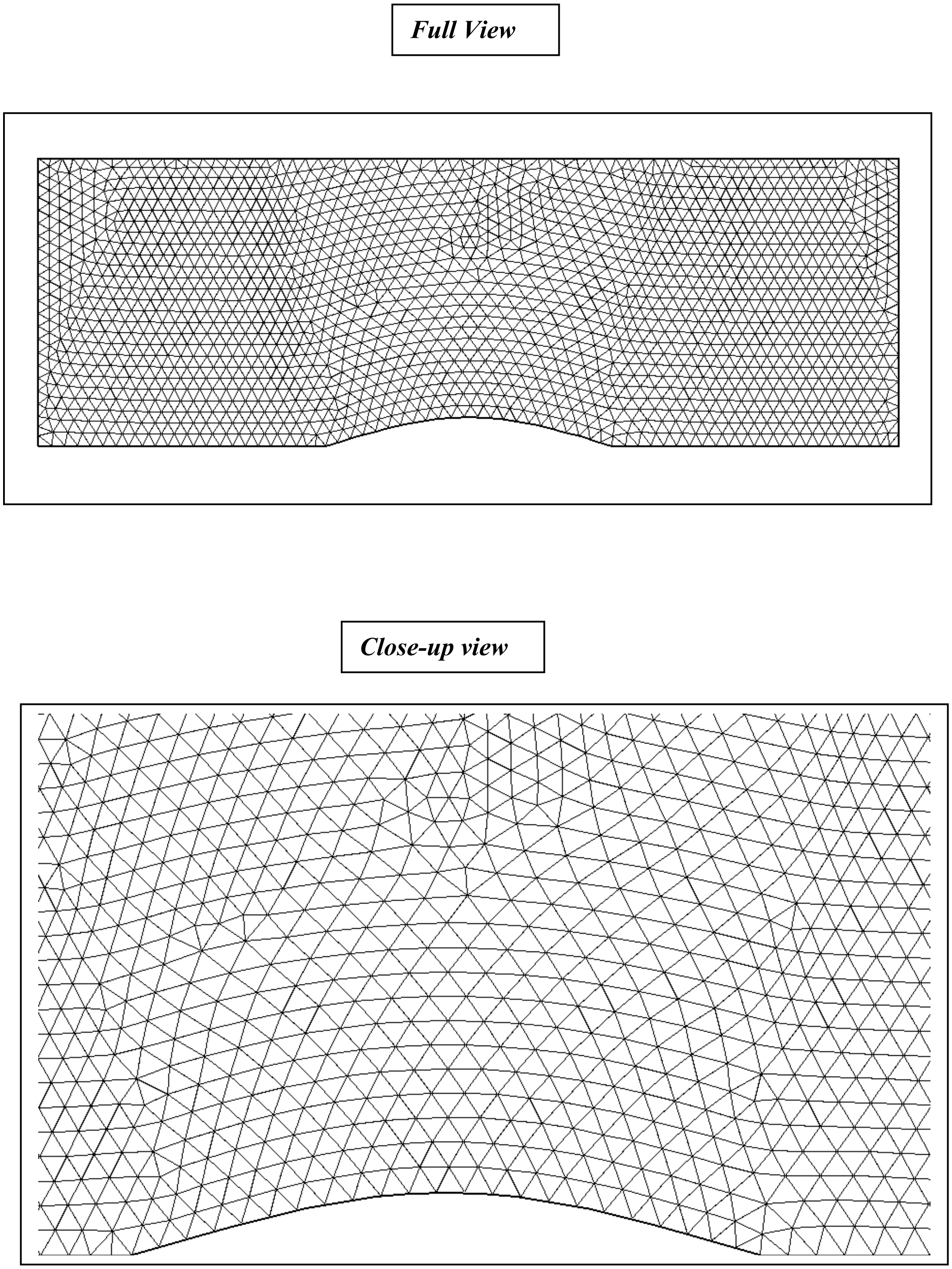
Euler Solution
min, max

Density = 0.4670, 1.930
Abs (V) = 0.7688, 1.150
Energy = 0.8878, 0.9926

Adjoint Solution
min, max

Density = -0.04049, 0.3088
Abs (V) = 0.00000, 0.6039
Energy = -0.07096, 0.2989

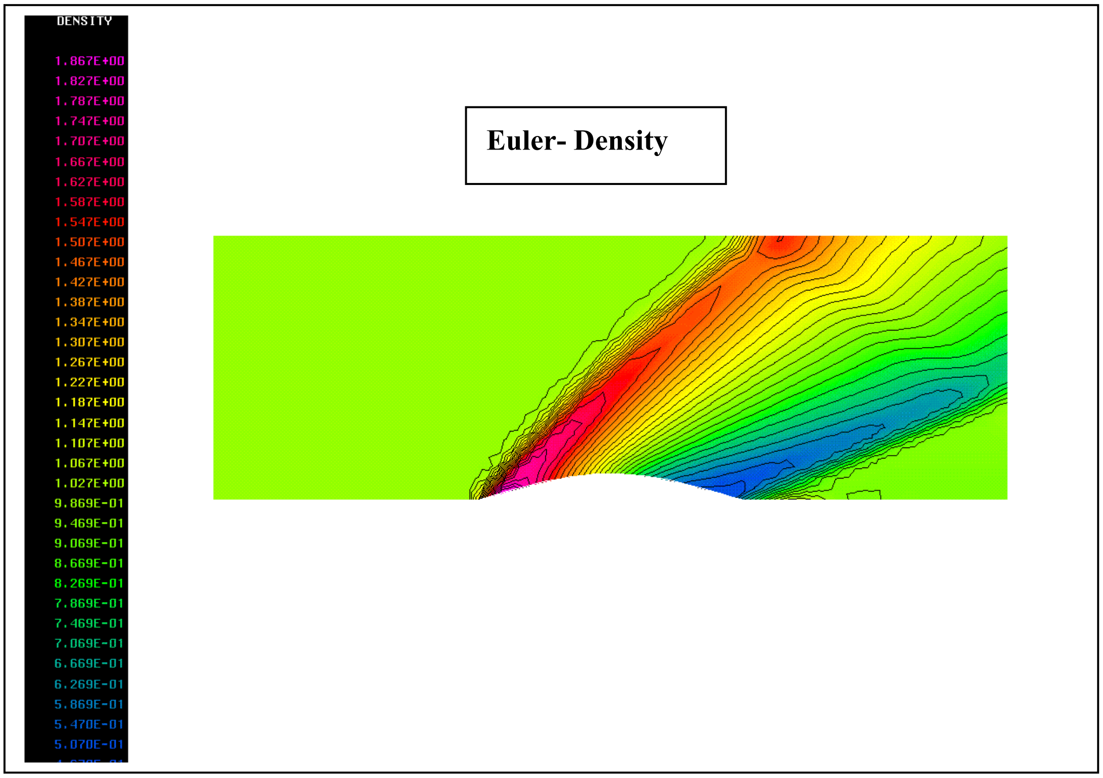
Fig. 8.1-1 Geometry and Flow Conditions for Symmetric Airfoil



Full View

Close-up view

Fig. 8.1-2 Mesh for Symmetric Airfoil



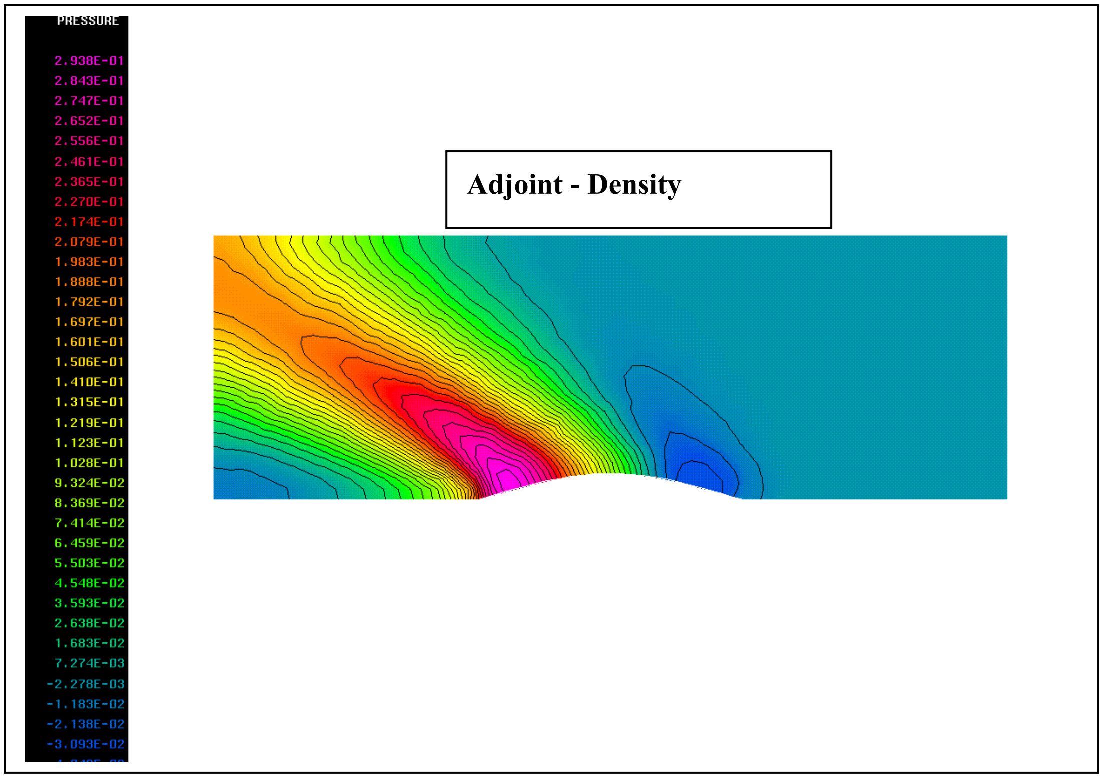
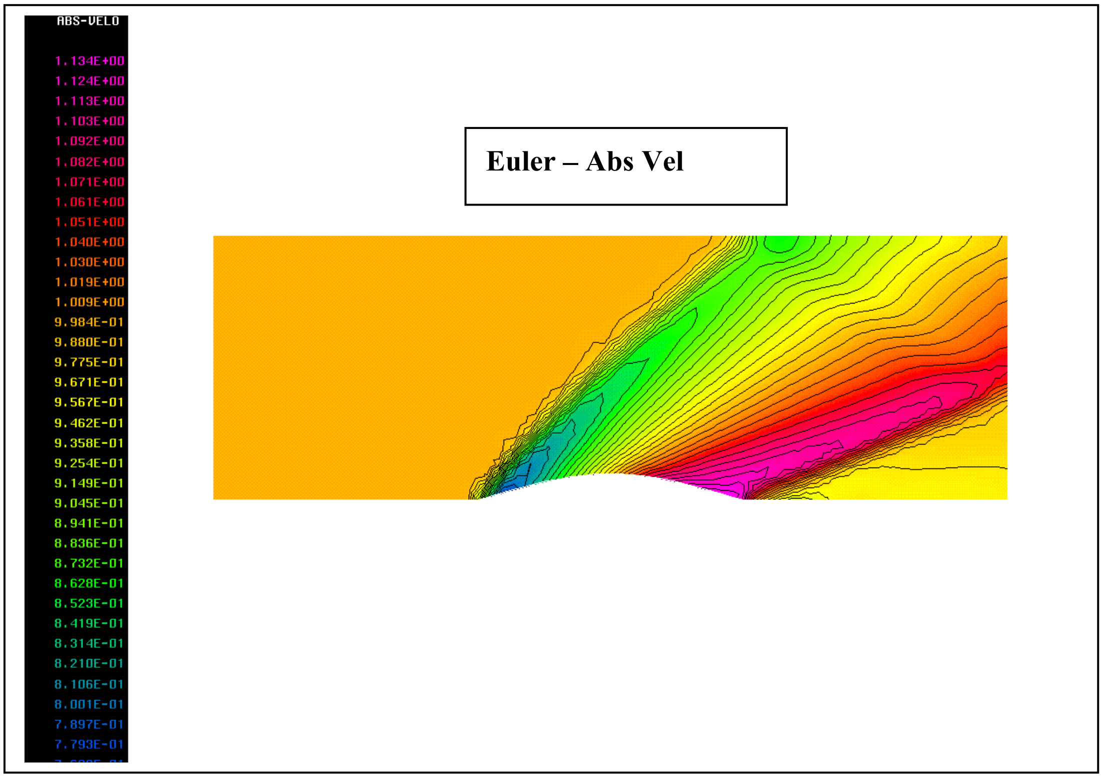


Fig. 8.1-3 Density Contours for Symmetric Airfoil



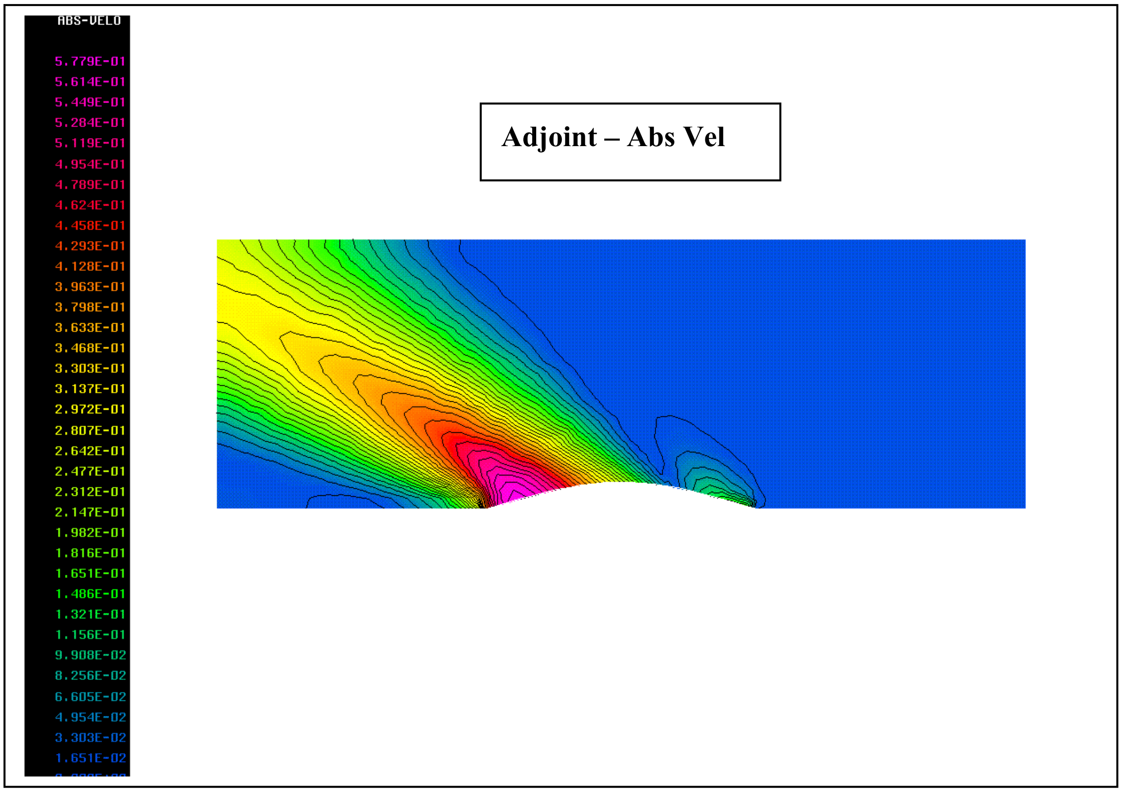
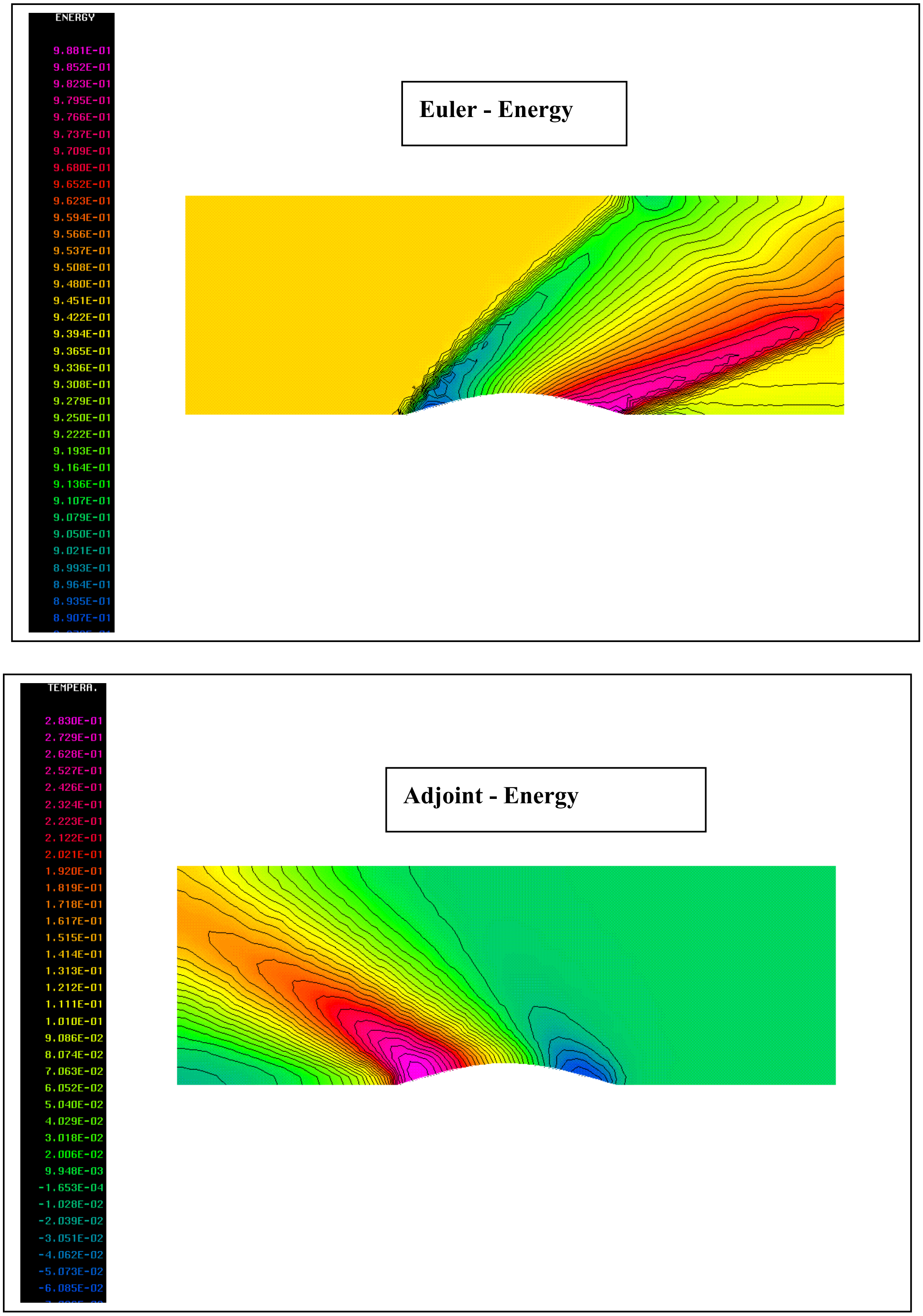
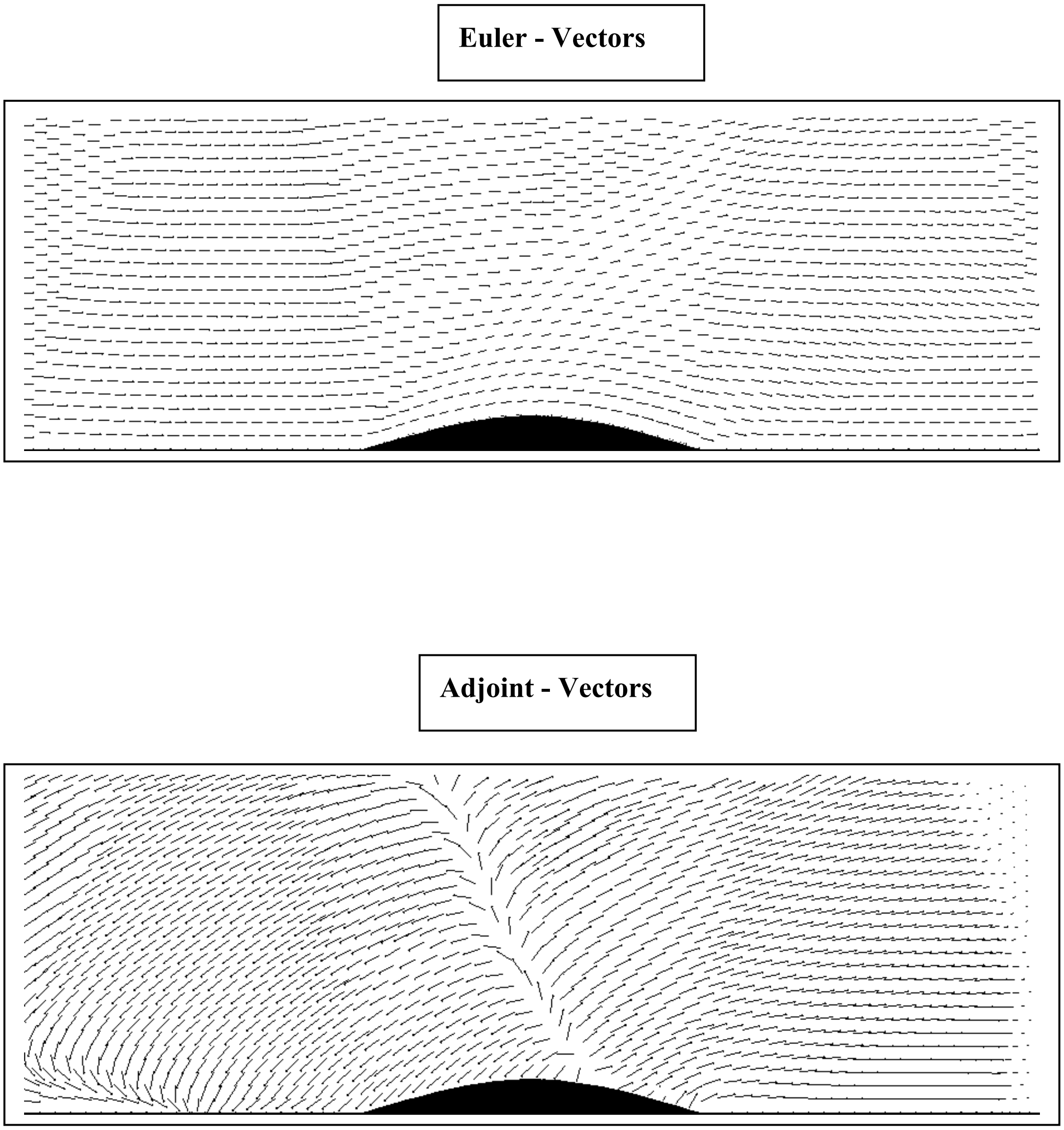


Fig. 8.1-4 Absolute Velocity Contours for Symmetric Airfoil


Euler - Energy

Adjoint - Energy

Fig. 8.1-5 Energy Contours for Symmetric Airfoil



Euler - Vectors

Adjoint - Vectors

Fig. 8.1-6 Velocity Vectors for Symmetric Airfoil

8.2 Validation for Airfoil

The analysis begins with a NACA0012 airfoil at Mach 2 and zero angle of attack, and the design optimization is directed toward a minimum drag airfoil shape. Inviscid (Euler) flow is assumed for this analysis. The chord length is held constant by fixing the leading and trailing edge points, and cross sectional area of the airfoil is held constant as the shape is changed. The initial shape is shown in Fig. 8.2-1.

The change in airfoil shape as optimization is reached is shown in Fig. 8.2-2. The minimum drag shape is a symmetric airfoil shape, as it should be, shown in RED, Finally, the computed airfoil drag coefficient is shown in Fig. 8.3-3 as the airfoil shape changes through the successive design steps. Convergence to the optimum shape is achieved in about eight design steps. The drag was reduced by about 50 %. The pressure contours on the initial shape and on the final shape age shown in Fig. 8.3-4.

These computations were made using a 2 GHz Pentium 4 Personal Computer under the Red Hat Linux operating system and Intel FORTRAN compiler. A computational mesh with 4937 mesh points and 21986 elements was used in the computations. The flow analysis used 130 implicit local time steps with a Courant number of 10 and required 40 cpu seconds of run time per design step. The adjoint analysis used 500 iterations and required 117 cpu seconds of run time per design step. The design gradient analysis required 2 cpu seconds of run time per design step to compute gradients with respect to mesh point movement. The design was called converged after 8 cycles. Overall, approximately 21 cpu minutes of computer run time were required to optimize the airfoil shape for minimum inviscid drag.

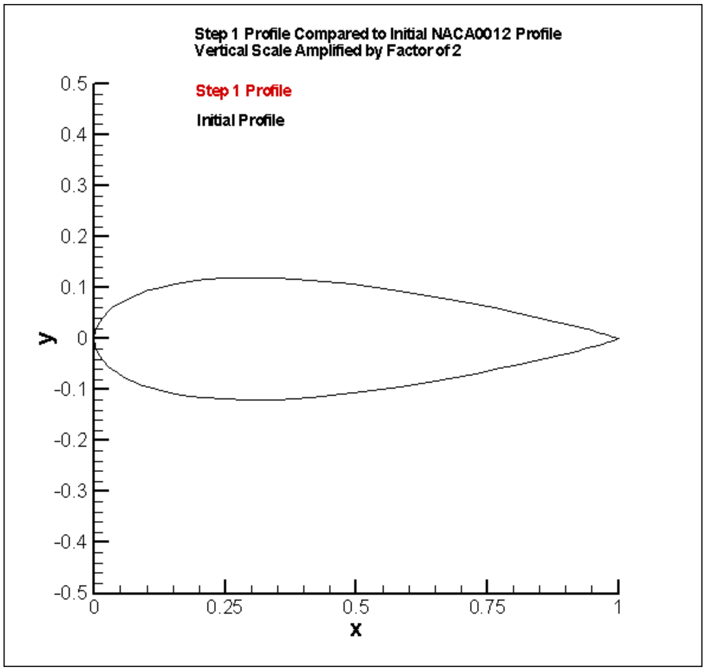


Fig. 8.2-1 Initial Airfoil Shape – NACA 0012

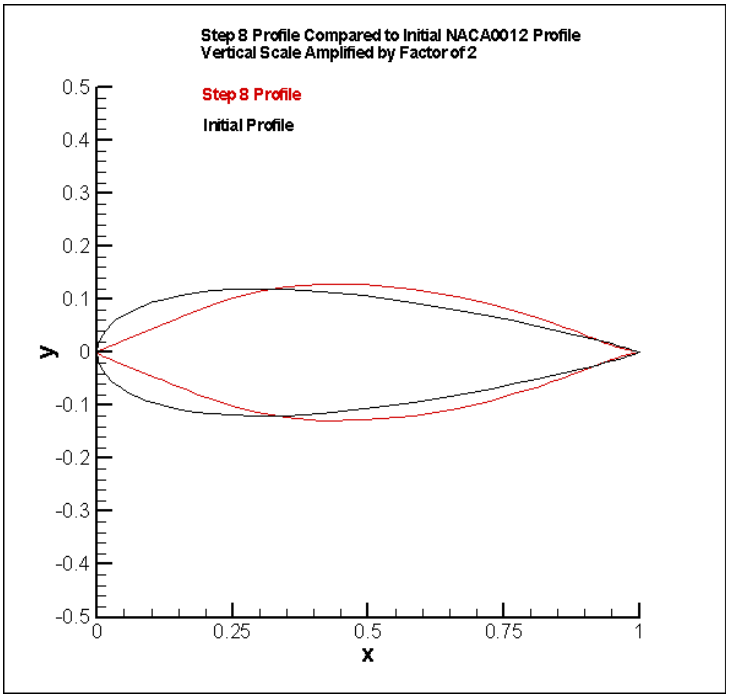


Fig. 8.2-2 Initial and Optimized Airfoil Shapes

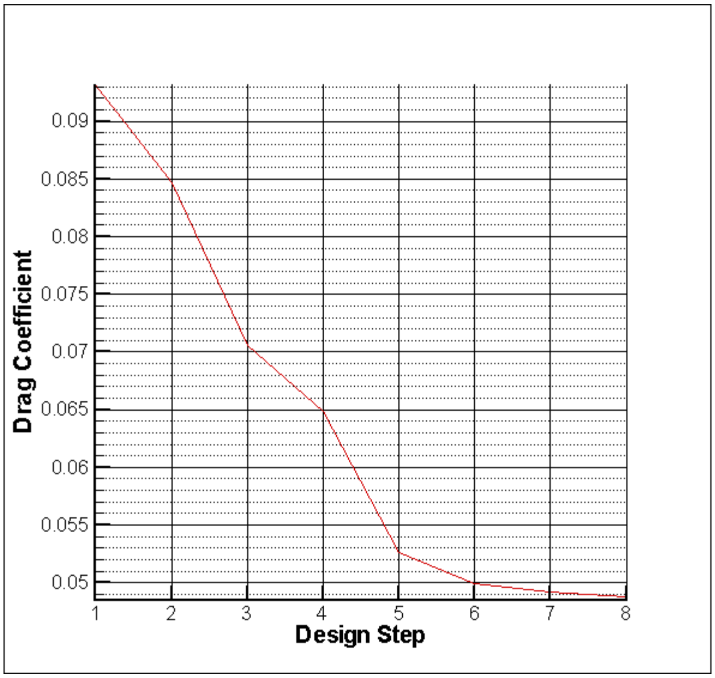
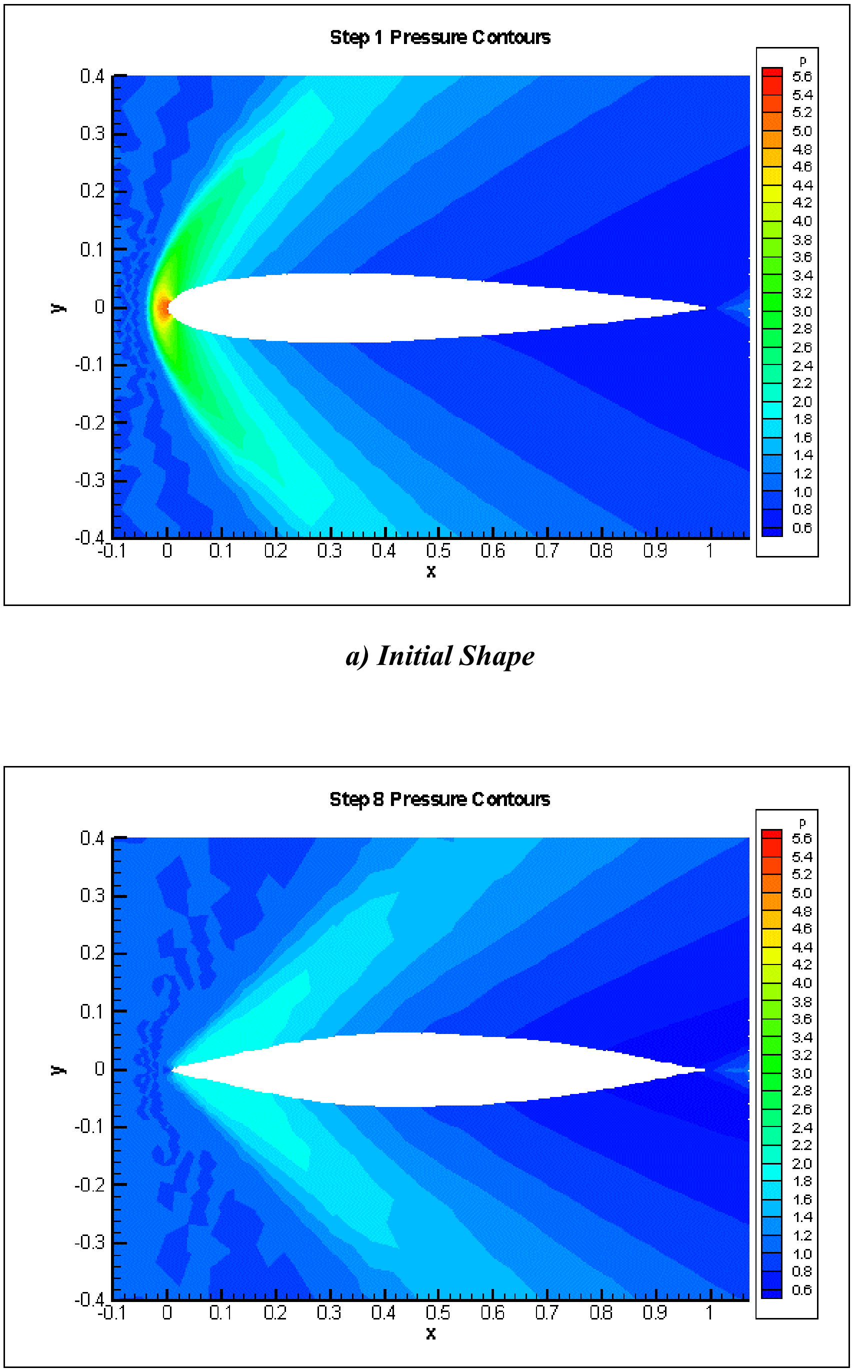


Fig. 8.2-3 Drag Reduction vs. Design Iteration



a) Initial Shape

b) Optimized Shape

Fig. 8.2-4 Pressure Contours on Initial and Optimized Airfoil Case

8.3 Validation for Nozzle

An optimization study was performed to maximize thrust for a 2D supersonic nozzle with an exit-to-throat area ratio of 3.0. The initial bell-shaped nozzle shape is shown in Fig. 8.3-1. The two end points are held fixed, and the two interior points are allowed to move in the vertical direction only. Moving these intermediate points during the optimization process reshapes the contour of the nozzle. The nozzle thrust consists of the sum of the momentum flow rate across the nozzle throat plane, which remains fixed, and the net force on the nozzle boundary in the forward direction, which varies with nozzle shape. The y-coordinates of the two interior points defining the nozzle contour shape are the two design parameters being varied to maximize the net force in the forward direction.

The thrust vs. design cycle is shown in Fig. 8.3-2. We concluded the run manually after 8 design cycles.

A plot of the meshes at design cycles 1, 4, 6 and 8 are shown in Fig. 8.3-3.The pressure contours for the bell-shaped nozzle at these same design iteration are shown in Fig. 8.3-4. Similar plots for the Mach contours are shown in Fig. 8.3-5.

Note that the optimized shape is close to a straight-line configuration.

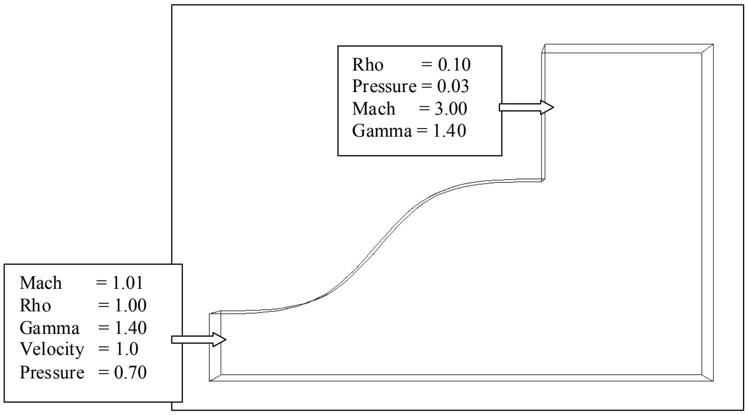


Fig. 8.3-1 Nozzle Configuration for Optimization

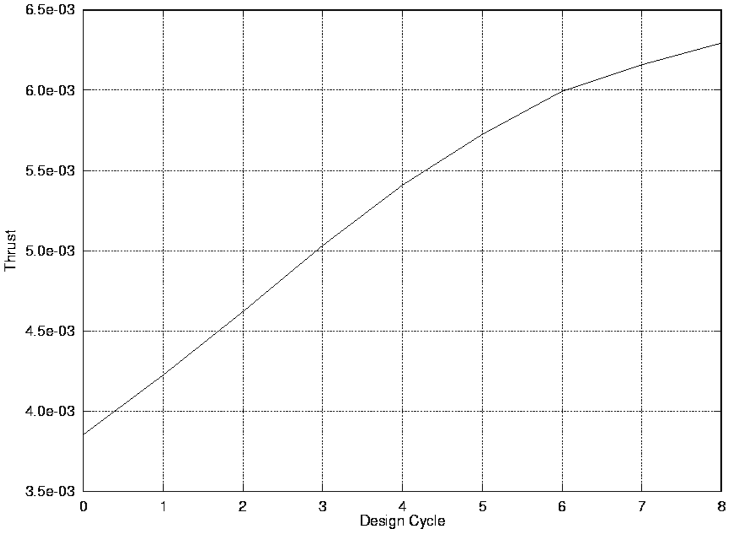


Fig. 8.3-2 Thrust Increase vs. Design Iteration for Nozzle Case

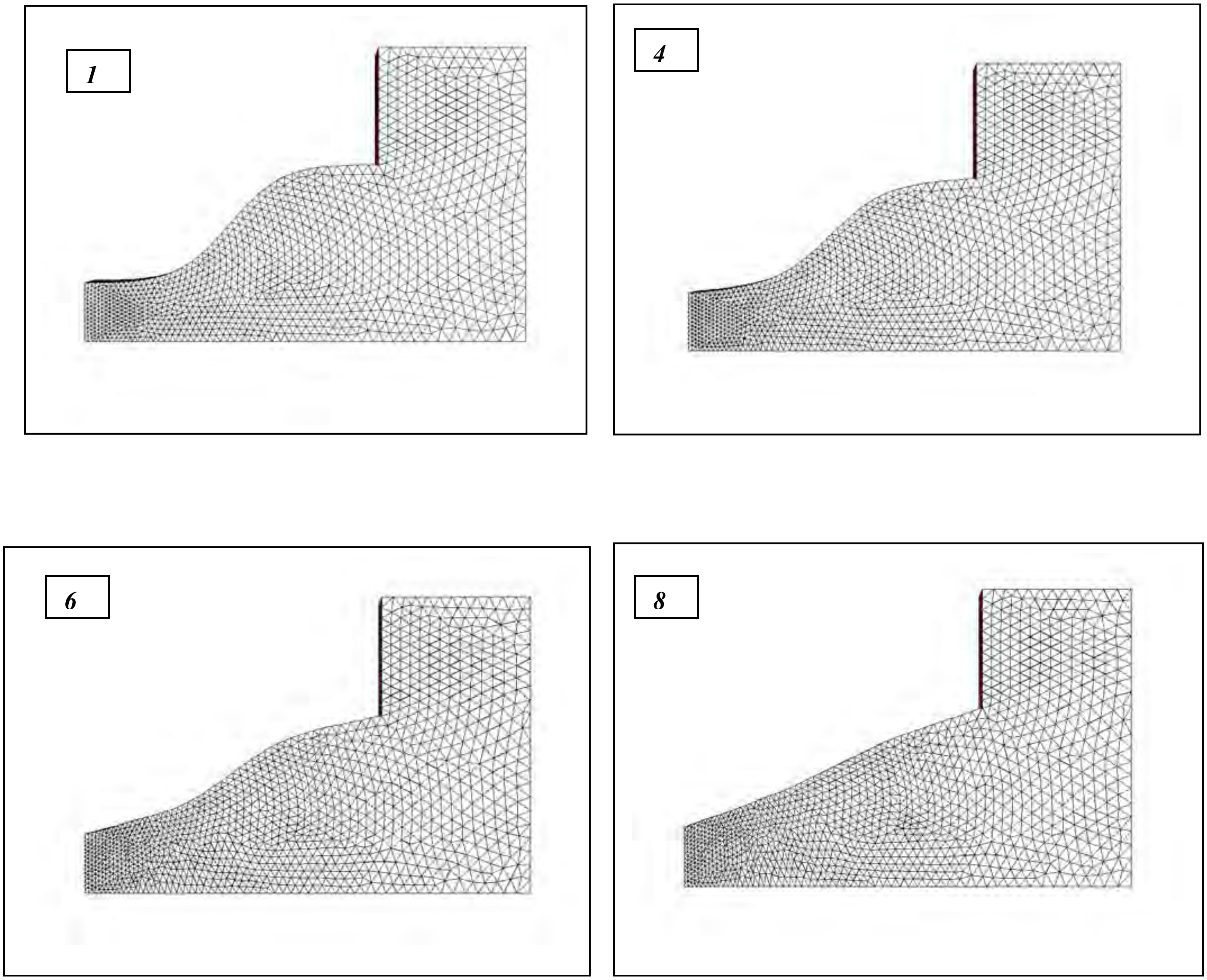


Fig. 8.3-3 Mesh at 4 Design Iterations

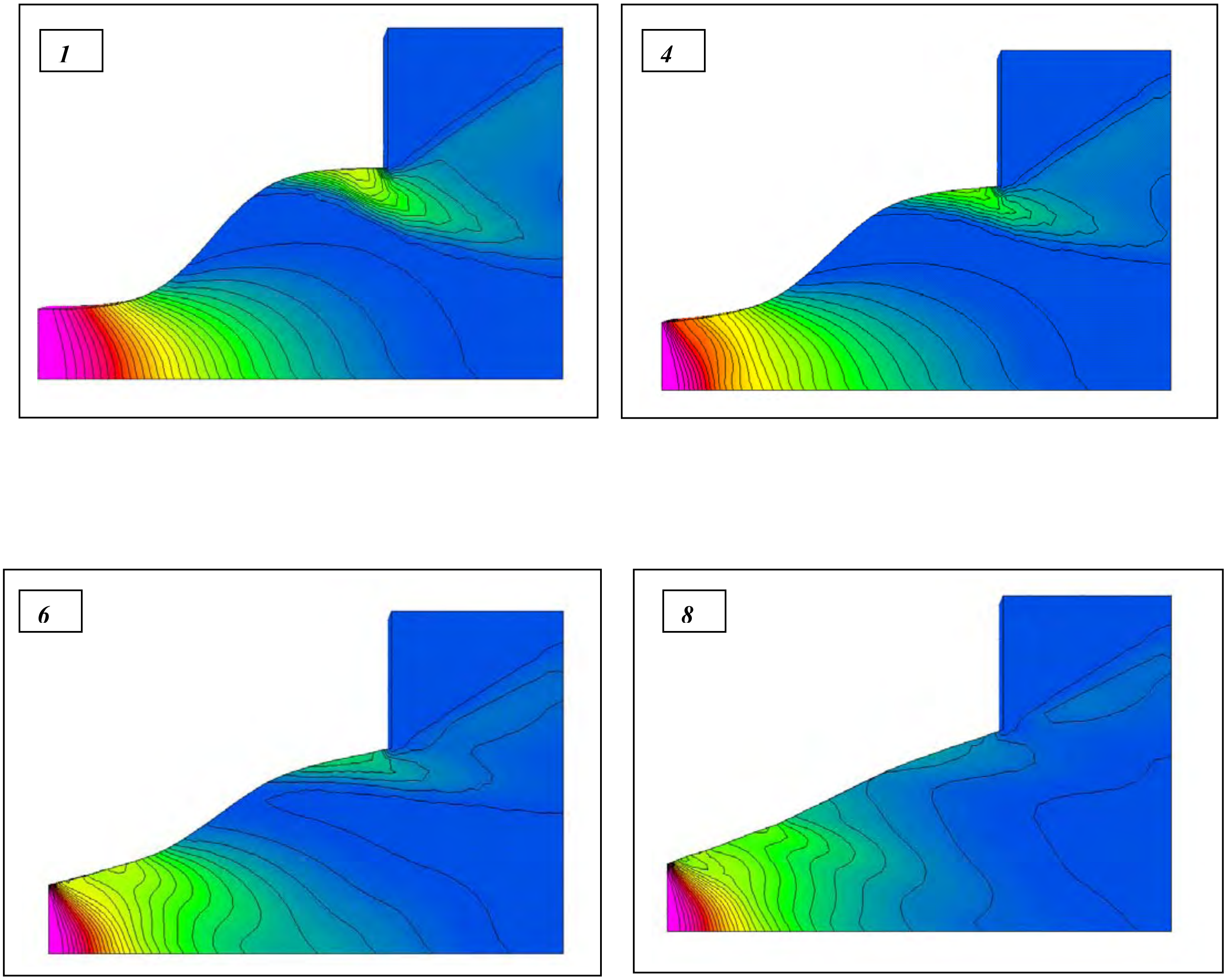


Fig. 8.3-4 Pressure Contours at 4 Design Iterations

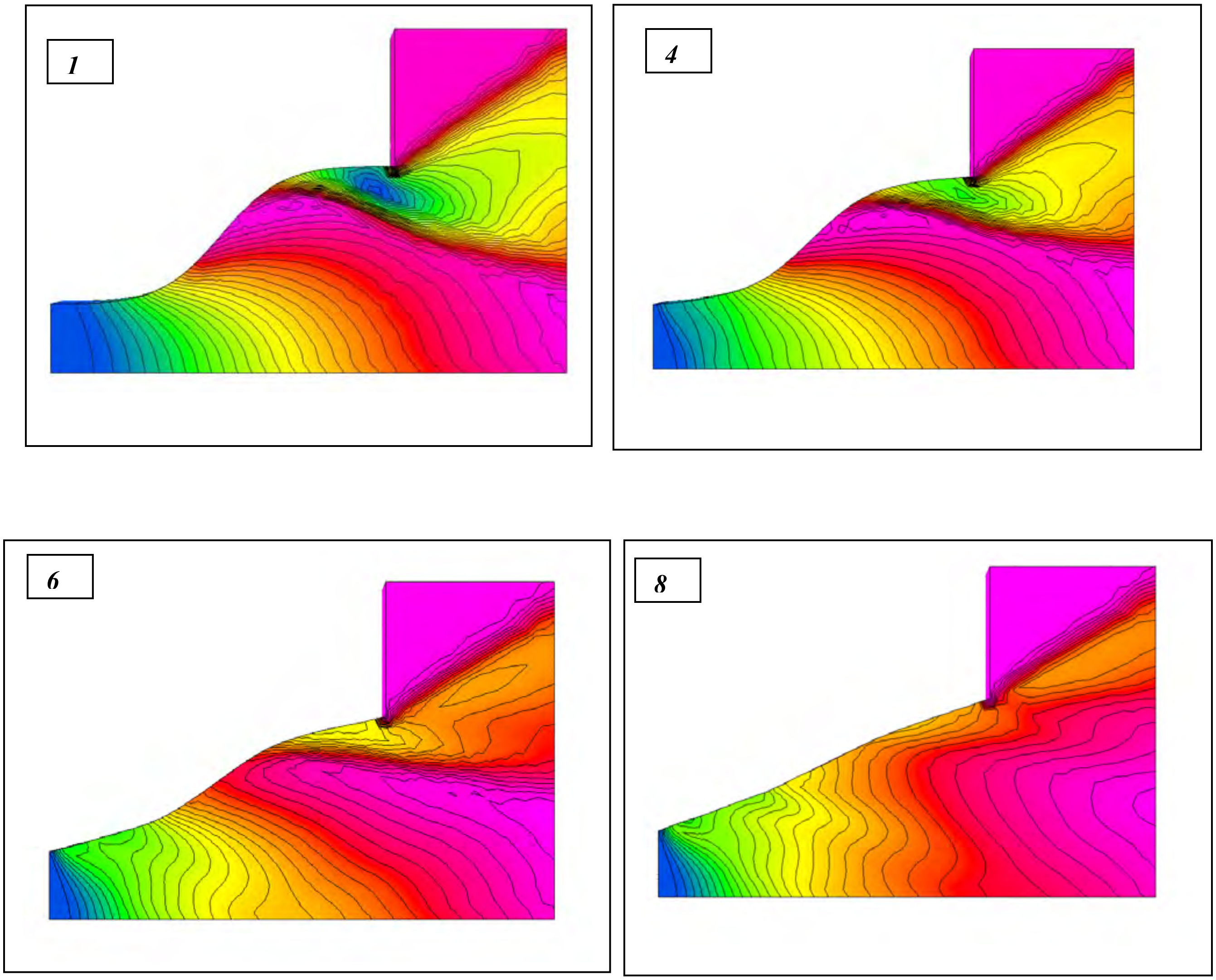


Fig. 8.3-4 Mach Contours at 4 Design Iterations

8.4 Generic Airbreathing Vehicle

The objective for this case is to optimize engine performance for a generic hypersonic air-breathing aircraft with a scramjet engine inlet. The vehicle geometry and computational domain and mesh are shown in Fig. 8.4-1. Symmetry allows modeling half the overall geometry with one bounding surface being the symmetry plane.

For this test case for the hypersonic aircraft, we optimize on mass capture for air flow into the inlet of the engine. The optimization process is being limited to reshaping the contour of the vehicle forebody between the leading edge and the engine inlet. The line from the leading edge to the inlet is defined by the end points, which remain fixed, and two intermediate points, which are allowed to move during an optimization step. A B-spline is fitted through these points to define the forebody contour. With each change in vehicle shape, a new computational mesh is created, and the flow around the vehicle and through the engine is recomputed. In particular, the mass flow rate through the inlet plane is computed. This is the object function that is to be optimized for this case.

Flow fields have been computed for the vehicle at Mach 6 and 30 km altitude. This case is for air flow only with no fuel injection or combustion.

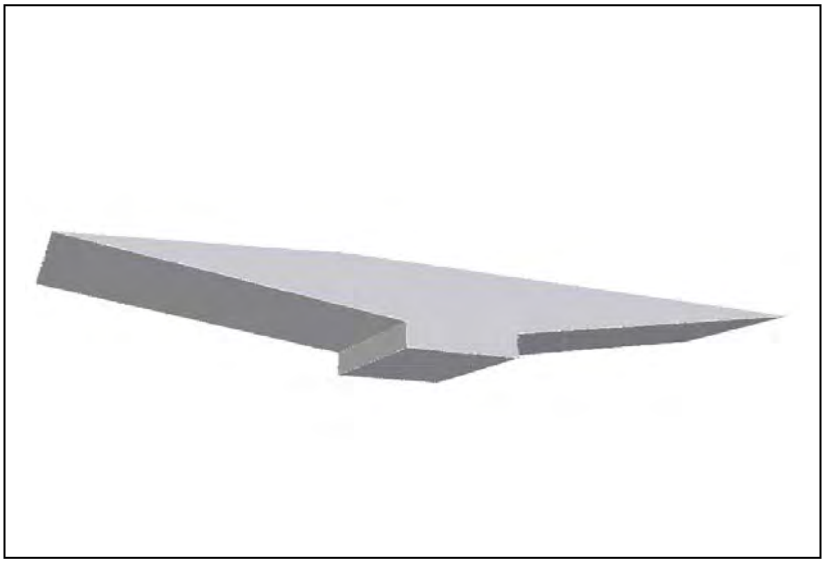
The computational mesh over the vehicle and symmetry plane surfaces is shown in Fig. 8.4-1 including a close up view of the engine mesh.

The final pressure field over the full plane, after optimization, is shown in Fig. 8.4-2.

The changes in forebody profile with each optimization step are shown in Fig. 8.4-3. The deviation from the initial profile is amplified by a factor of 5 for enhanced visualization of the small changes in forebody shape.

The initial mass capture rate prior to optimization is 7.28 kg/s. The capture rate is improved steadily with each of the first four optimization steps, apparently approaching an optimum improvement. This is then followed by a final rather dramatic improvement to 7.95 kg/s. At this point, the optimization process ends due to a lack of further improvements with changes in forebody geometry. The optimized forebody design provides a 9.2 % improvement in air mass capture.

The pressure and temperature contours in the engine are plotted for the initial and final forebody shapes in Fig. 8.4-4.



Freestream conditions

Mach = 6 at 30 km
Density = 0.004826
Velocity = 2519.2
Pressure = 341.9
Gamma = 1.4
Alpha = 2 deg.

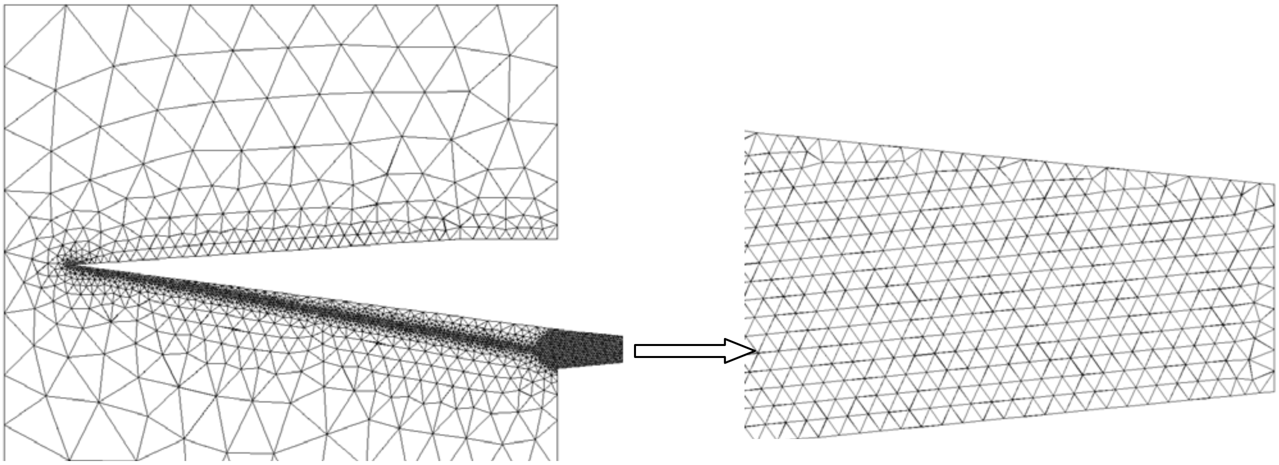


Fig. 8.4-1 Generic Airbreathing Vehicle – configuration, flow conditions and mesh

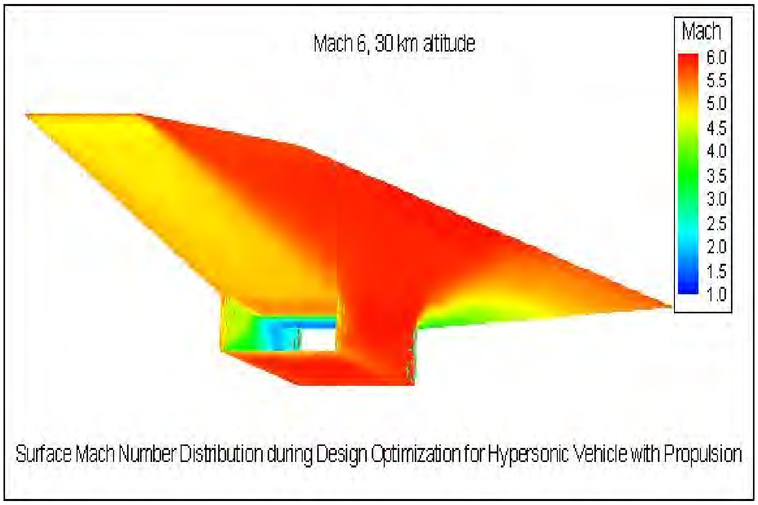

Mach 6, 30 km altitude
Mach
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Surface Mach Number Distribution during Design Optimization for Hypersonic Vehicle with Propulsion

Fig. 8.4-2 Mach Contours on Optimized Forebody

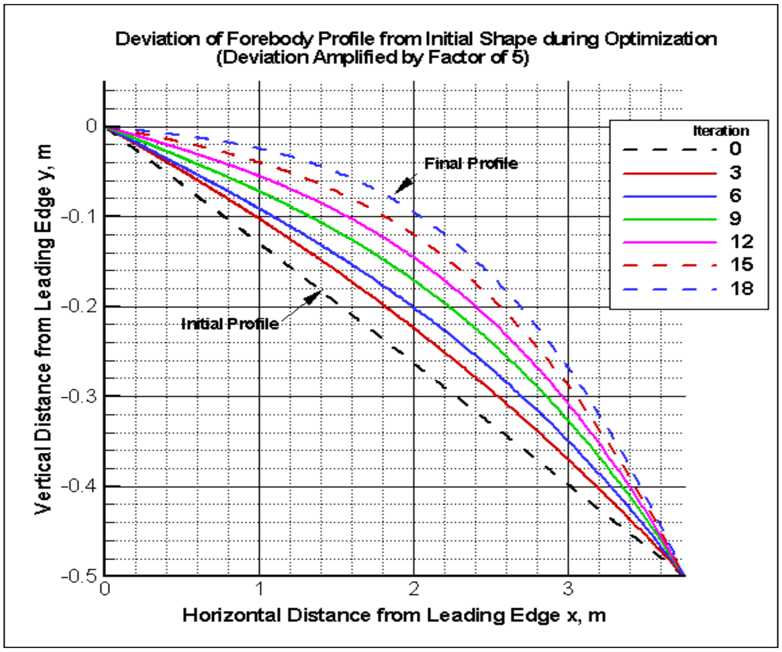


Fig. 8.4-3 Forebody Profile vs. Design Iteration

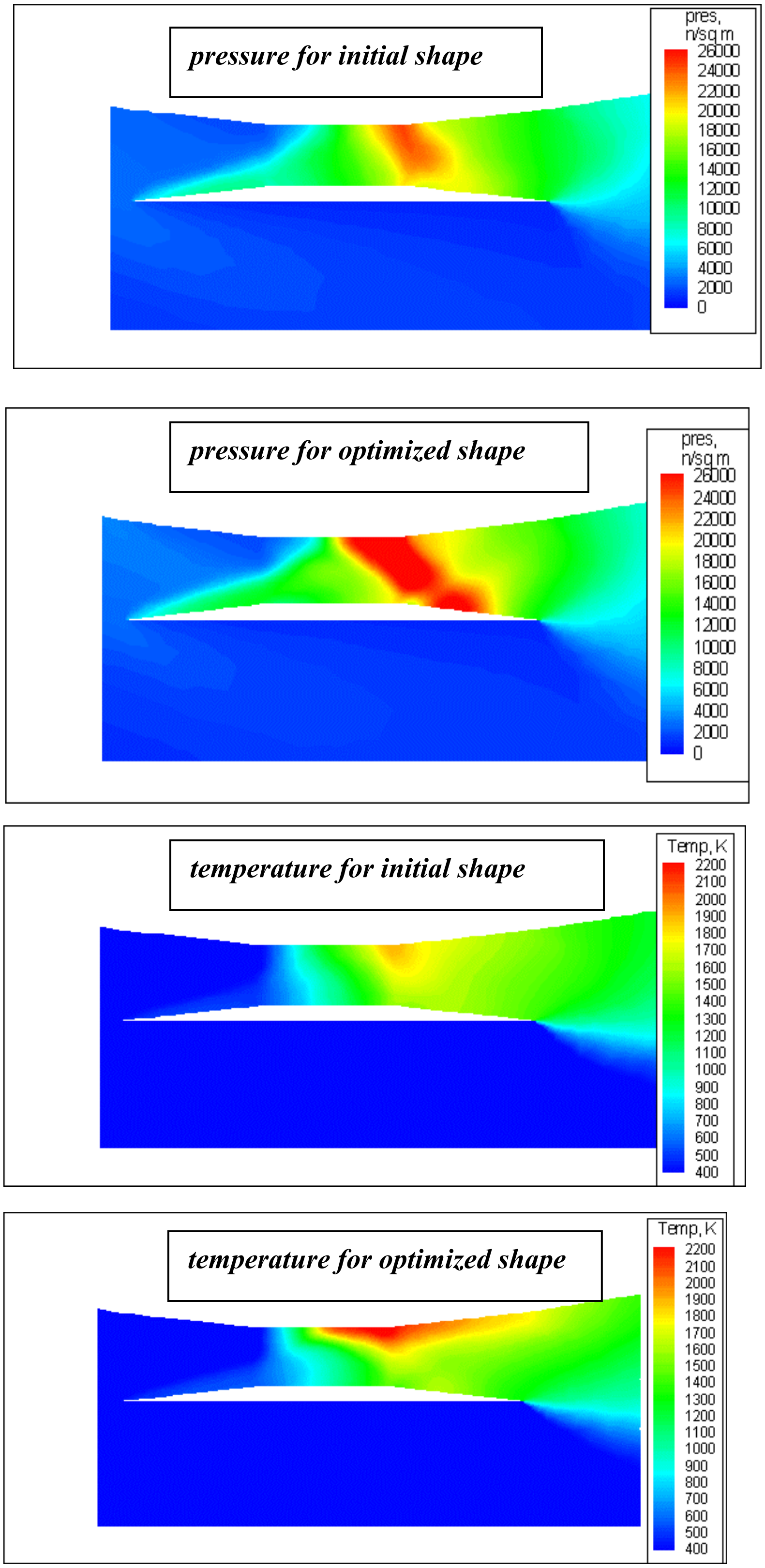


Fig. 8.4-4 Contours in Engine for Initial and Optimized Vehicle

8.5 Generic HRRLS Configuration

Consider again our generic HRRLS flyer as shown in the top of Fig. 8.5-1. The flight conditions are Mach = 2.5 and 5 deg angle of attack and inviscid flow is assumed.

For simplicity to illustrate the adjoint method on an HRRLS type vehicle, we restricted the optimization to the nose design only. The objective function is to reduce the aerodynamic drag on the nose. The method was halted after 5 iterations producing the shape shown in the bottom of Fig. 8.5-1. A much sharper nose shape results and reduced the drag by 15 percent. A close up view of the nose region shapes are given in Fig. 8.5-2.

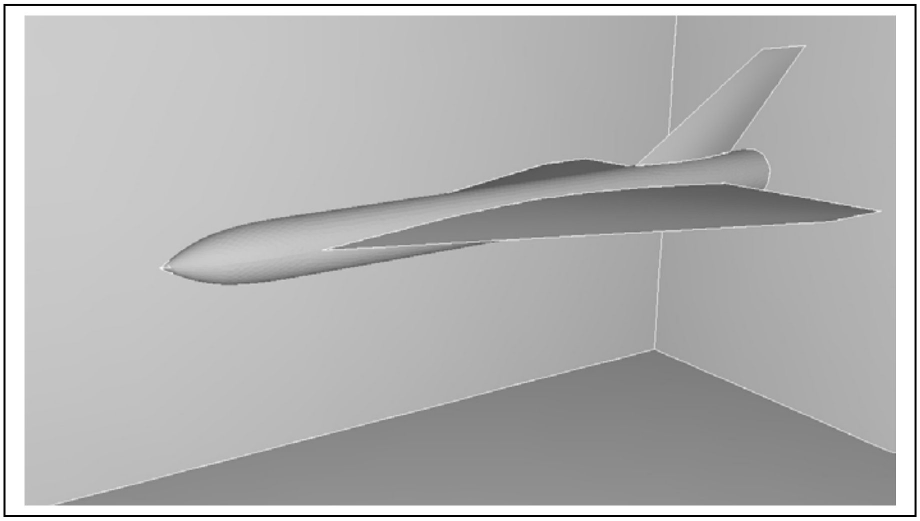
The steady state pressure contours for the original and final nose design are compared in Fig. 8.5-3.

Next, the adjoint solutions are illustrated for the final HRRLS flyer design. Fig. 8.5-5 shows steady state density contours on the top part and the adjoint density on the bottom part of the figure. The adjoint velocity vectors are compared in Fig. 8.5-5 for the initial and final designs. A close up of the nose region is shown in Fig. 8.5-6 for the adjoint velocity vectors.

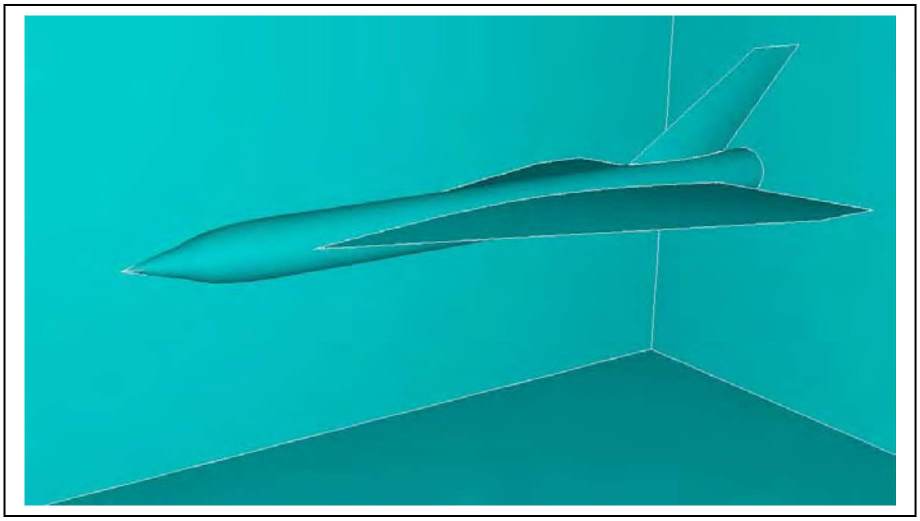
8.6 References

[1] Jameson, A., N. A. Pierce, L. Martinelli, Optimum Aerodynamic Design Using the Navier-Stokes Equations, AIAA Paper 97-0101 (1997).

[2] http://www.scs.gmu.edu/~rlohner/pages/publications/fluidstructint.html

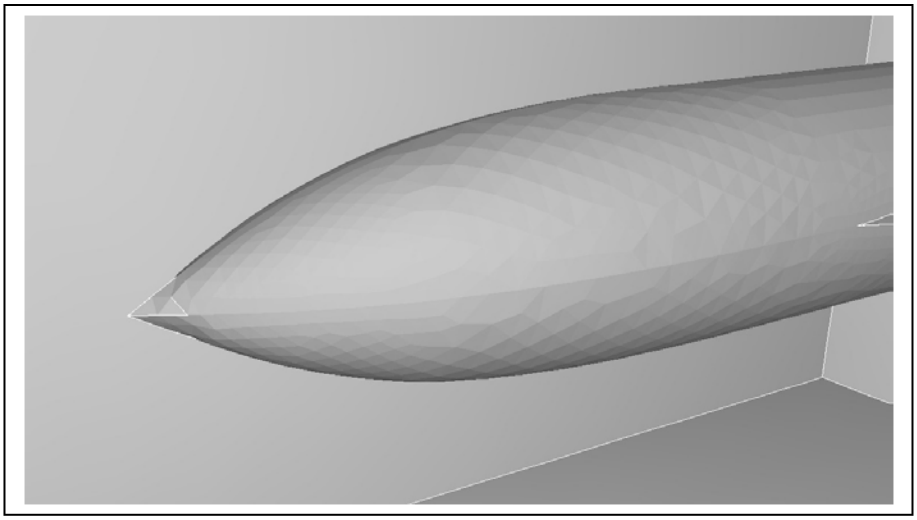


original vehicle shape

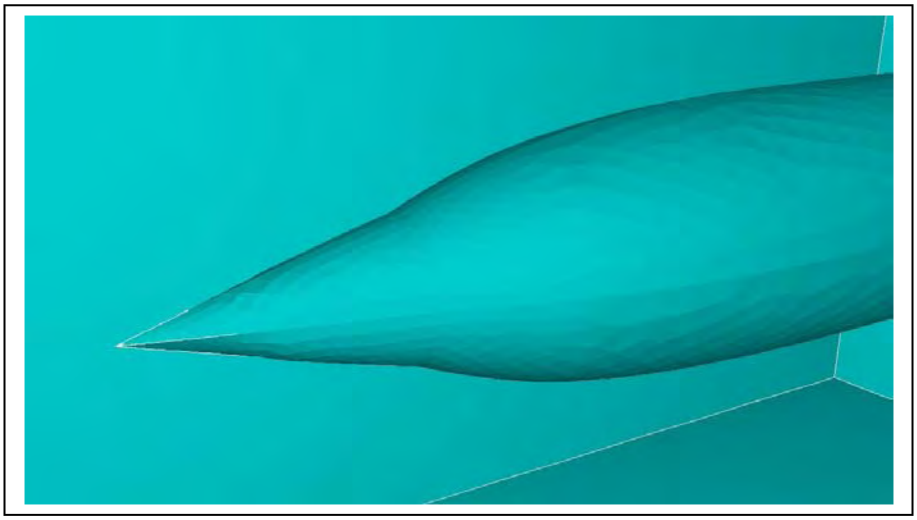


final vehicle shape

Fig. 8.5-1 Original and Final Design Shape

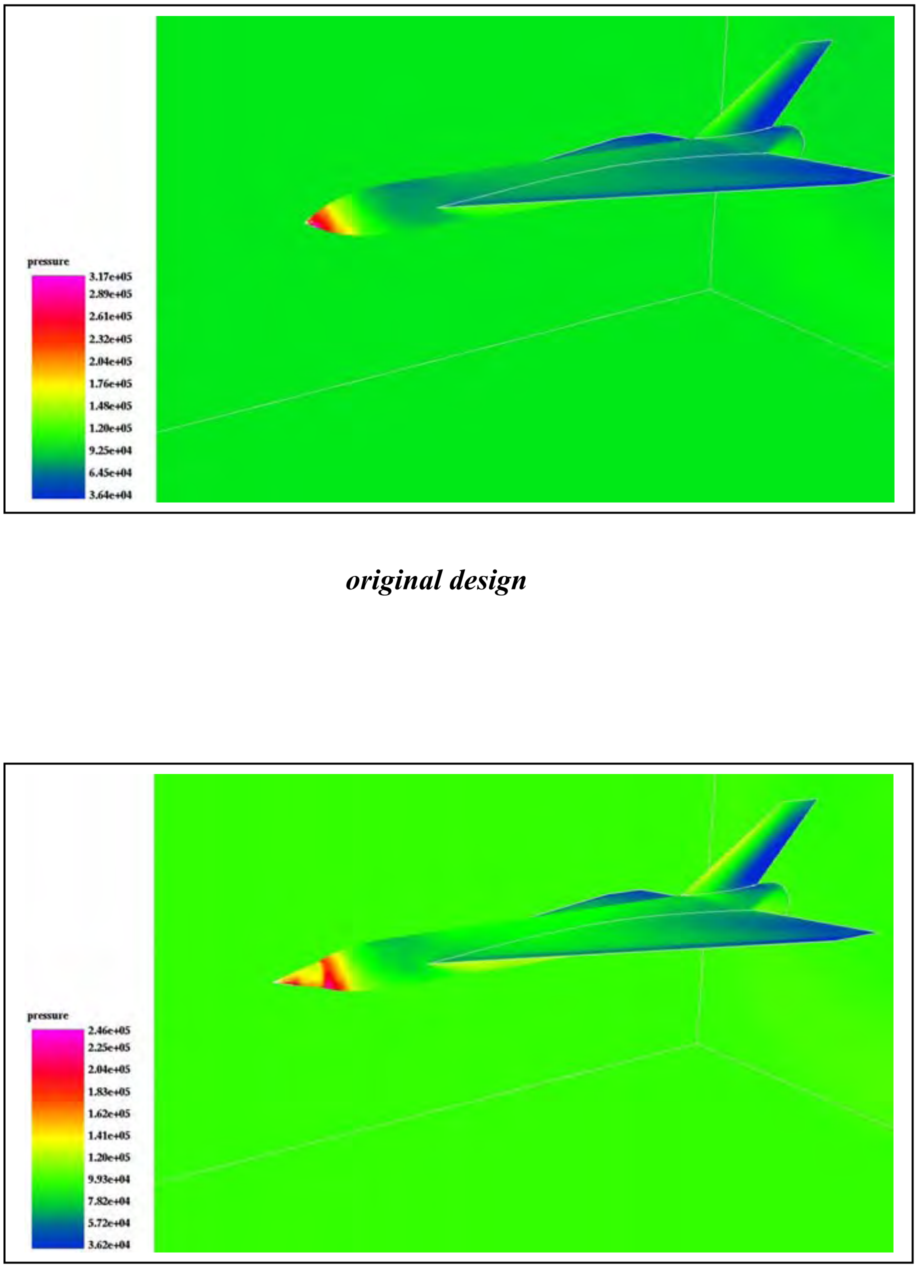


original nose region design



final nose region design

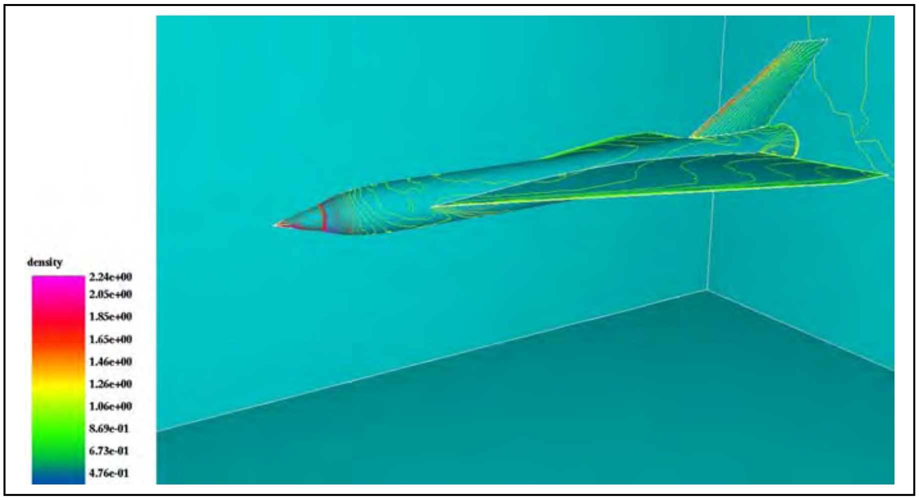
Fig. 8.5-2 Initial and Final Design – Close up of Nose Region



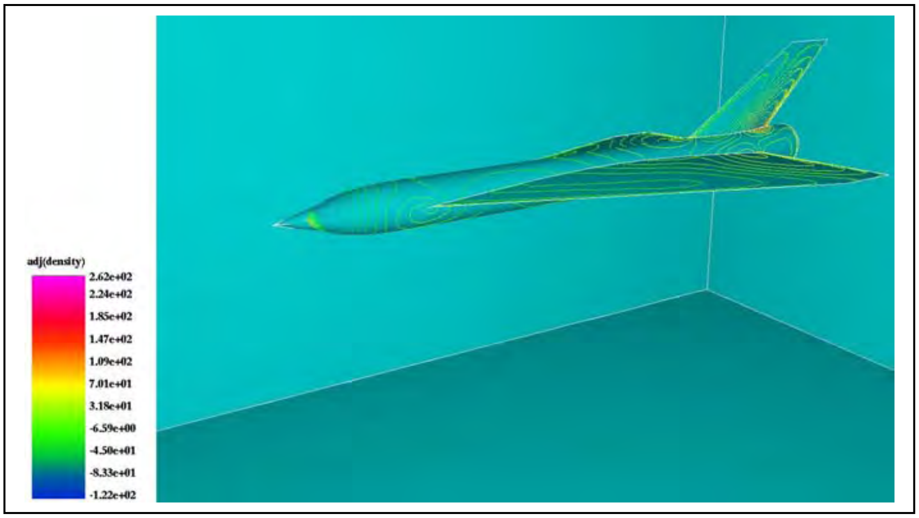
original design

final design

Fig. 8.5-3 Pressure Contours for Initial and Final Nose Design

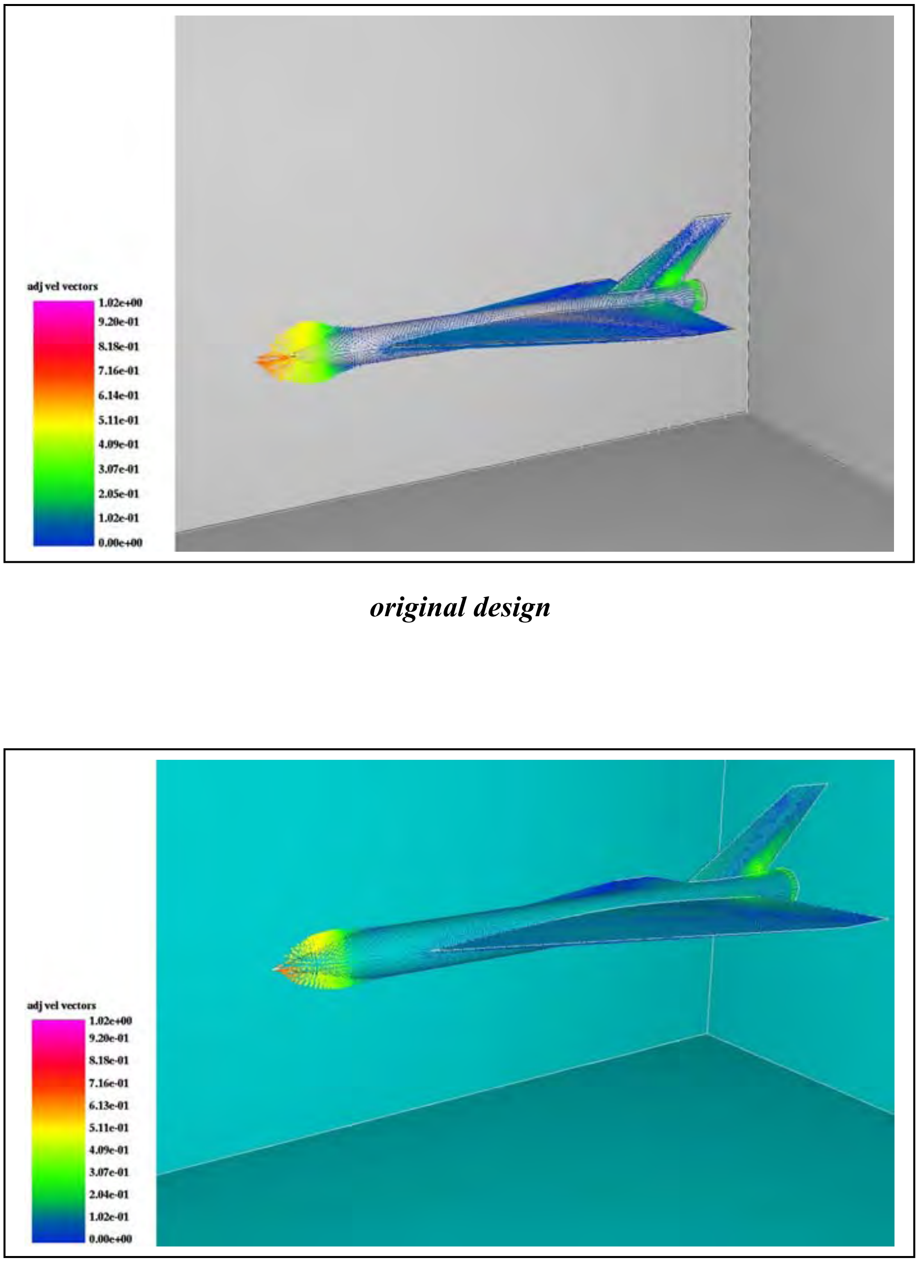


density



adjoint density

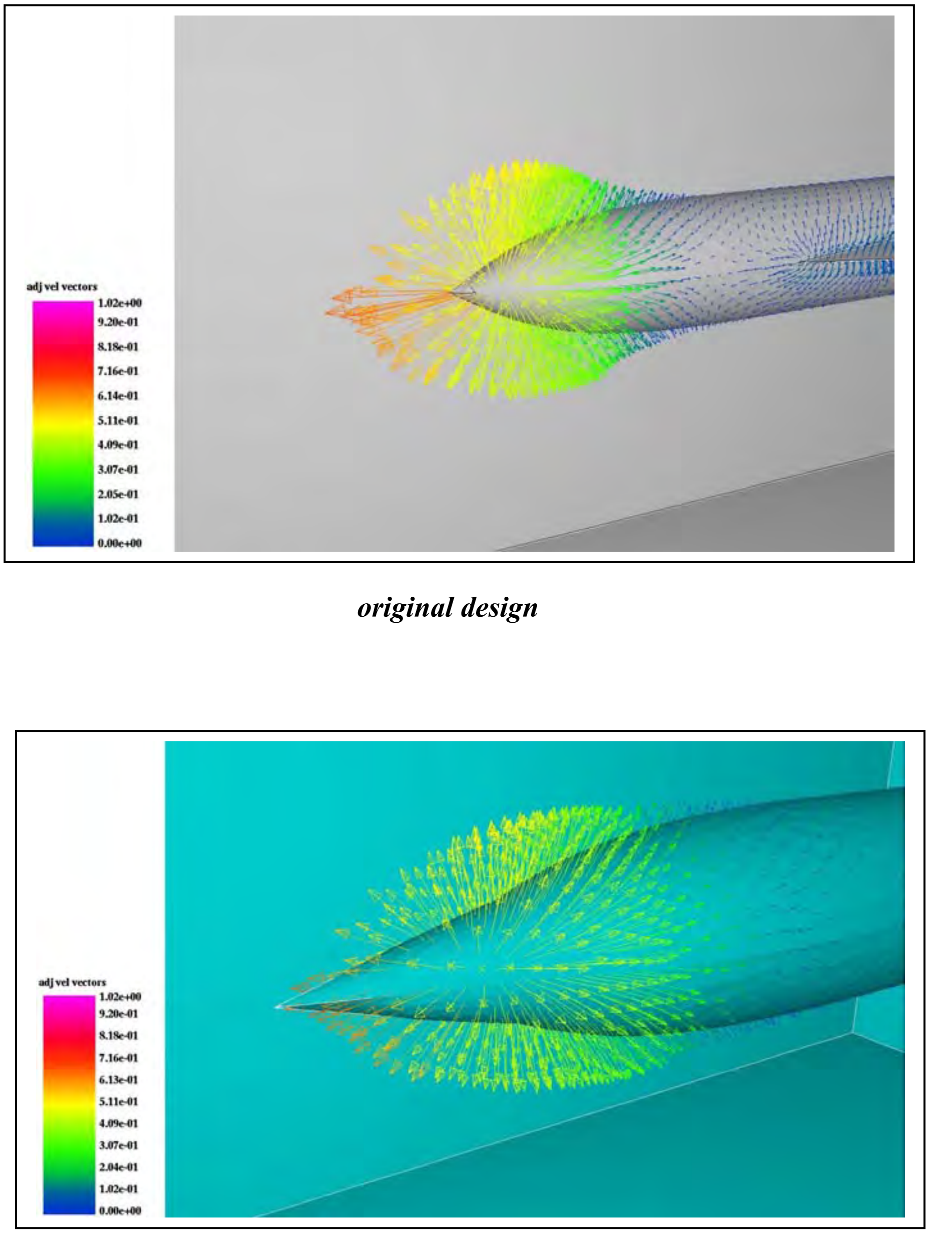
Fig. 8.5-4 Density and Adjoint Density on Final Design



original design

final design

Fig. 8.5-5 Adjoint Velocity Vectors on Initial and Final Design



original design

final design

Fig. 8.5-6 Adjoint Velocity Vectors on Original and Final Design
Enlargement of Nose Region

9.0 UNCERTAINTY ANALYSIS

9.1 Information Grids

The advent of modern Personal Computers and Giga-bit internet has led to the emergence of so-called information grids. With further advances in pre-processing, the expectation on engineers will be not a single computational mechanics run per day, but many hundreds of runs. This trend is fully compatible with engineering practice, where the "once-only" run is the exception.

In the vast majority of cases, engineers are required to scope a parameter space in order to identify the critical parameters and dimensions of a problem, or to study the sensitivity of the design with respect to slightly different flow conditions (angle of attack, temperature, etc) or geometry. This allows us to do many runs of a similar kind with slight variations in the input conditions. Once the mesh for the individual cases has been generated, the flow and initial forces/positions are varied, and the resulting loads and deformations are determined. An uncertainty analysis of the output can then be done.

The only thing that changes from run to run is a few parameters. Given that the CPU requirement of each one of these runs is similar, lends itself naturally to parallel processing on an information grid. The data obtained from the different input variations can then be used to carry out an uncertainty analysis. The internet continues to expand in speed and capacity at a rate superior to that of Moore's law. This implies that sending jobs to be run remotely, and retrieving results, is going to become easier as time progresses. We can envision scenarios where the user only operates a small PC, yet conducts run across the nation on the currently available supercomputers. Missing at present are the protocols and ease of use to make such an environment a reality. The information grid idea is specifically directed towards this shortcoming.

9.2 Uncertainty Methods

In an AIAA special issue in 1998, Oberkampf and Blottner presented a detailed discussion of sources of uncertainty from simulation of physical phenomena [1]. In 2002, Oberkampf, et al presented a paper on "Mathematical Representation of Uncertainty" [2]. The book by Ghanem and Spanos [3] gives details of the spectral approach to stochastic finite elements. A paper by Walters, et al [4] describes in detail the mathematics of "Uncertainty Analysis for Fluid Mechanics". His paper reviews uncertainty analysis methods and their application to fundamental problems in fluid dynamics. The focus in on error analysis for CFD type methods, but the same mathematics applies to the fluid-structure interaction problems discussed above. Walters gives the following summary [4]: "One of the primary reasons for the increased interest in uncertainty management is its application in risk-based design methods. The structures community and the dynamics and controls discipline have a long history in uncertainty analysis while the CFD community is a relative newcomer. This is due to the relative infancy of the discipline, but mainly to the large costs of CFD simulations.

The uncertainty analysis methods which we have investigated can be classified as follows:

Deterministic	Probabilistic
Interval Analysis, Sensitivity Derivatives:	Monte Carlo, Moment Methods, Polynomial Chaos

Interval Analysis

This is the simplest and most straight-forward method. The idea is to perform operations at input intervals which contain the set of all possible values of the driving input data. The output will then consist of all possible outcomes from the input parameters. The interval results then represent the worst case scenario and provide the maximum error bounds. This method can be directly implemented in a CFD solver virtually transparent to the user. The only problem is that the massive amounts of output data must be post-processed.

Sensitivity Derivatives

Sensitivity derivatives are discussed in detail by Dahlquist and Bjorck [5]. The method derives equations for a set of dependent variables in terms of a sum of squared errors of the independent variables. The errors are computed using "sensitivity" derivatives of the dependent variables times an error estimate. Experimental data comparisons have shown this method to yield the entire uncertainty band [4].

Monte Carlo

A good history of probabilistic Monte Carlo methods is given in the older book by Hammersley and Handscomb [6]. Applying this method to uncertainty analyses consists of the following three steps: 1) sample input random variables from their known probability density function, 2) compute the deterministic output for each sampled input, and 3) determine the statistics, mean, variance and skewness for each sampled input parameter. This Monte Carlo approach converges to the exact stochastic solution as the number of samples increases.

Moment Methods

The moment method has been used extensively by CFD researchers [7]. In this approach, approximations are obtained from truncated Taylor series expansions about an expected value of the input parameter. A first moment is just the value computed at the mean of the input parameter. The second moment is computed using an improved estimate of the mean and a derivative of the dependent variable. This method can be applied to multiple random variables with variance and covariance values and correlation coefficients being obtained as output.

Polynomial Chaos

Ghanem et al [8, 9] present a good discussion of the polynomial chaos method for uncertainty analysis. This method relies on a spectral representation of uncertainty. The basic idea is to decompose a random variable into separate deterministic and stochastic components. A polynomial basis, such as Hermite polynomials, is used to span the random space to be represented. The statistics of the random variables can be obtained for Gaussian and other distribution functions. The method offers the advantage of modeling both the deterministic and random parts of a function.

9.3 Sensitivity Derivative Example

Computational fluid dynamicists often study simplified problems which contain certain features of a more complicated situation but at lower cost and effort. Many such problems exist, but one that is particularly useful is the model non-linear convection-diffusion problem solved by Walters and Huyse [10] and referred to as the general Burgers equation. Based on specific choices of parameters in the equation, we can obtain linear or non-linear behavior. We consider the steady form of this equation, which in conservation law form is given by:

$$\frac{\partial f}{\partial x} - \frac{\partial}{\partial x}(\mu \frac{\partial u}{\partial x}) = 0$$

where the flux, f, is a non-linear function of u, and μ is the fluid viscosity. We analyzed the specific case shown below:

$$f = \mathrm{u}\ (1/2 - u)$$

Coleman and Steele [11] present a methodology which they call "General Uncertainty Analysis". The equations are actually sensitivity derivatives of the test function. If analytical relationships exist among the parameter space, then an analytical expression can be obtained by differentiating this relation with respect to the stochastic variable. Most of the real world cases do not exhibit these simple relations, and in any case they would be very lengthy and tedious. In our case, we choose the viscosity to be stochastic, and thus the derivatives of velocity with respect to viscosity are the key derivatives. We used the Non-Linear Burgers equation to test this method. For generality, we computed these derivatives numerically by finite differences for varying values of μ, and over the x-range from -3 to +3.

Figure 9.1 shows the first and second sensitivity derivative for the Non-Linear burgers equation. Each increasing derivative is more oscillatory. These plots are thus good for accessing the uncertainty of the velocity field as a function of the uncertainty in viscosity.

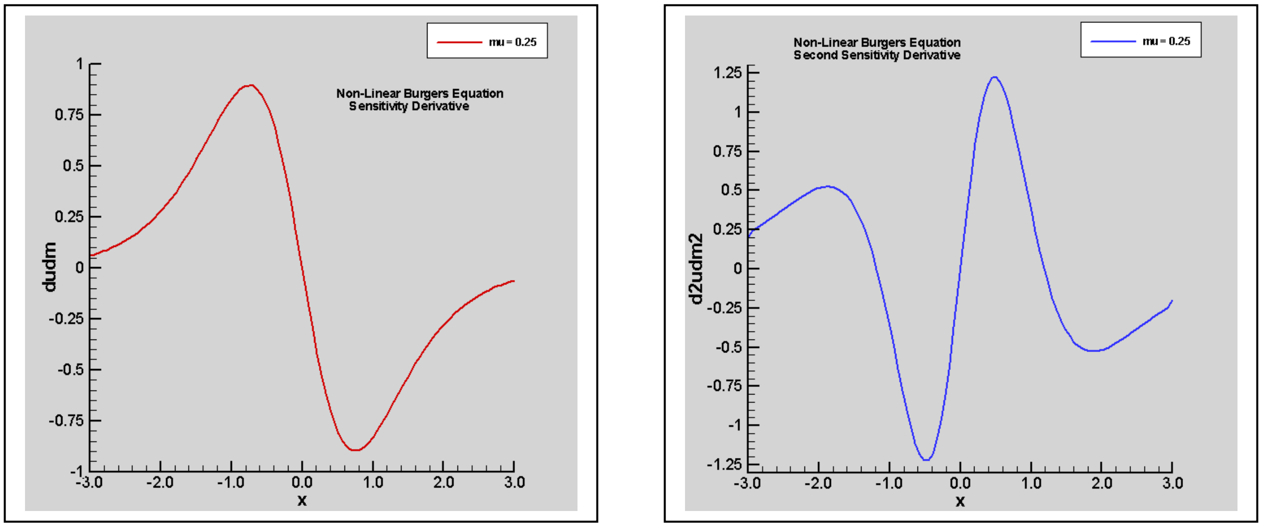


Fig. 9.1 First and Second Sensitivity Derivatives for the Nonlinear Burgers Equation

9.4 Generic HRRLS Vehicle

We have computed an example case for a generic HRRLS vehicle and applied a Monte Carlo method to computing uncertainty parameters. A summary of this case is presented here to illustrate the use of the methods that we are currently applying to the HRRLS effort.

The problem modeled is a generic vehicle shown in Fig. 9.2. An inviscid case was run at Mach = 2.5. We took as the stochastic variable, the angle of attack. A Gaussian distribution was generated for the angle of attack from -4.95 to 14.6. A total of 100 runs were made for this angle of attack input distribution. Figures 9.2 and 9.3 show density, pressure and Mach number at one representative of attack (for illustration) as line contours.

Figure 9.4 is a shaded plot of the pressure contours at this one angle of attack. Shown next in Fig. 9.4 is the computed lift and drag forces for each of these 100 runs.

A total of 100 runs were made to test the Monte Carlo method. Each CFD run completed in 2 minutes of CPU time. Total time was 2 minutes x 100 cases = ~ 200 minutes on the Pentium i7 PC machine.

The results are shown as histograms of the uncertainty distributions. The ordinate is the number of cases executed within a given band. Figure 9.5 shows the angle of attack distribution of for the 100 runs which we set. Note that the input uncertainty is Gaussian-like. The output uncertainty in lift force is shown in the histogram in the lower part of Fig. 9.5. Likewise, the histogram for drag force is shown in Fig. 9.5. Note that for these cases, the output uncertainty distributions look much like the angle of attack uncertainty. This is not always the case as shown in previous literature [12].

Table 9.1 is the summary statistics for the 100 Monte Carlo runs. The mean standard deviation, variance, and Kurtosis, and others are shown here for completeness. This type of information can now be obtained quickly, and on a desktop PC, such that many cases can be evaluated in a practical manner. Each configuration, flight condition, material type, and other parametric properties will have its own uncertainty domain from which to make an evaluation.

Finally, the lift and drag forces are plotted vs. angle of attack with percentage error bands in Fig. 9.7. For illustration, the lift is shown with 10% error bands and the drag with 1 % bands.

typical input flow conditions for each angle of attack

```
     dens           u-velo          v-velo          w-velo       pres
0.121300E+01   0.680000E+03  10.000000E+00   0.000000E+00   1.0e+5
```

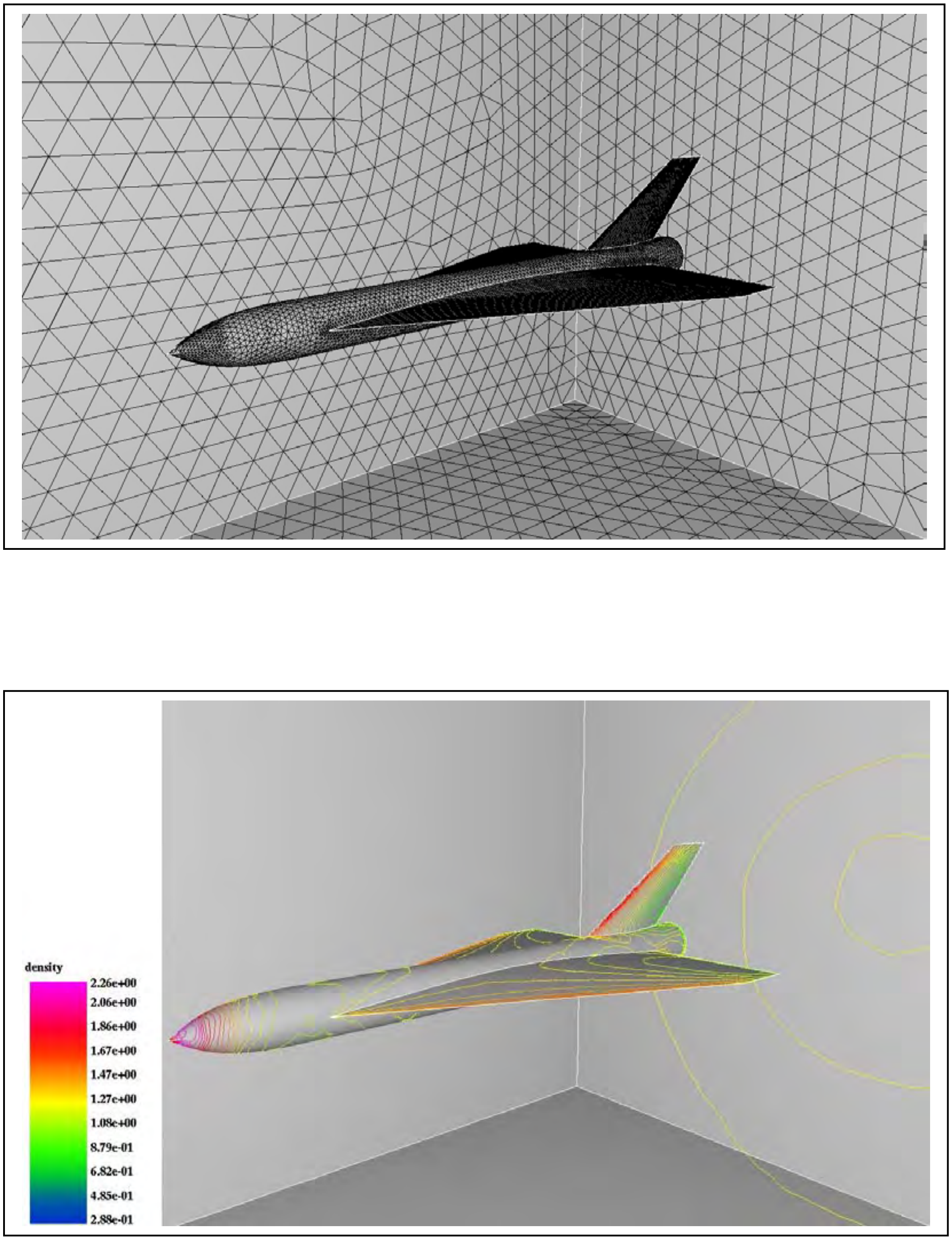


Fig. 9.2 Generic Configuration and Steady State Density at One Selected Angle of Attack

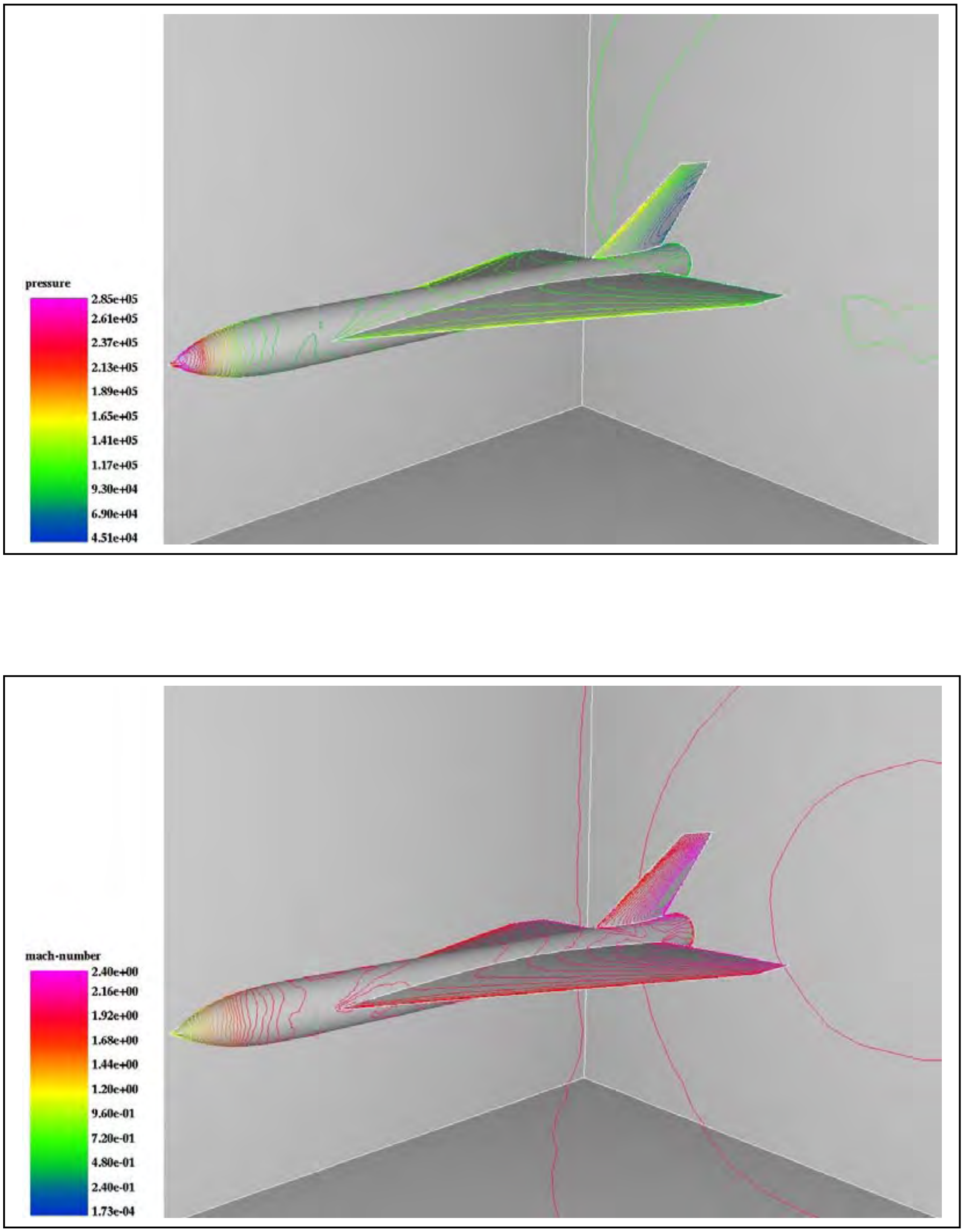


Fig. 9.3 Steady State Pressure and Mach Contours at One Selected Angle of Attack

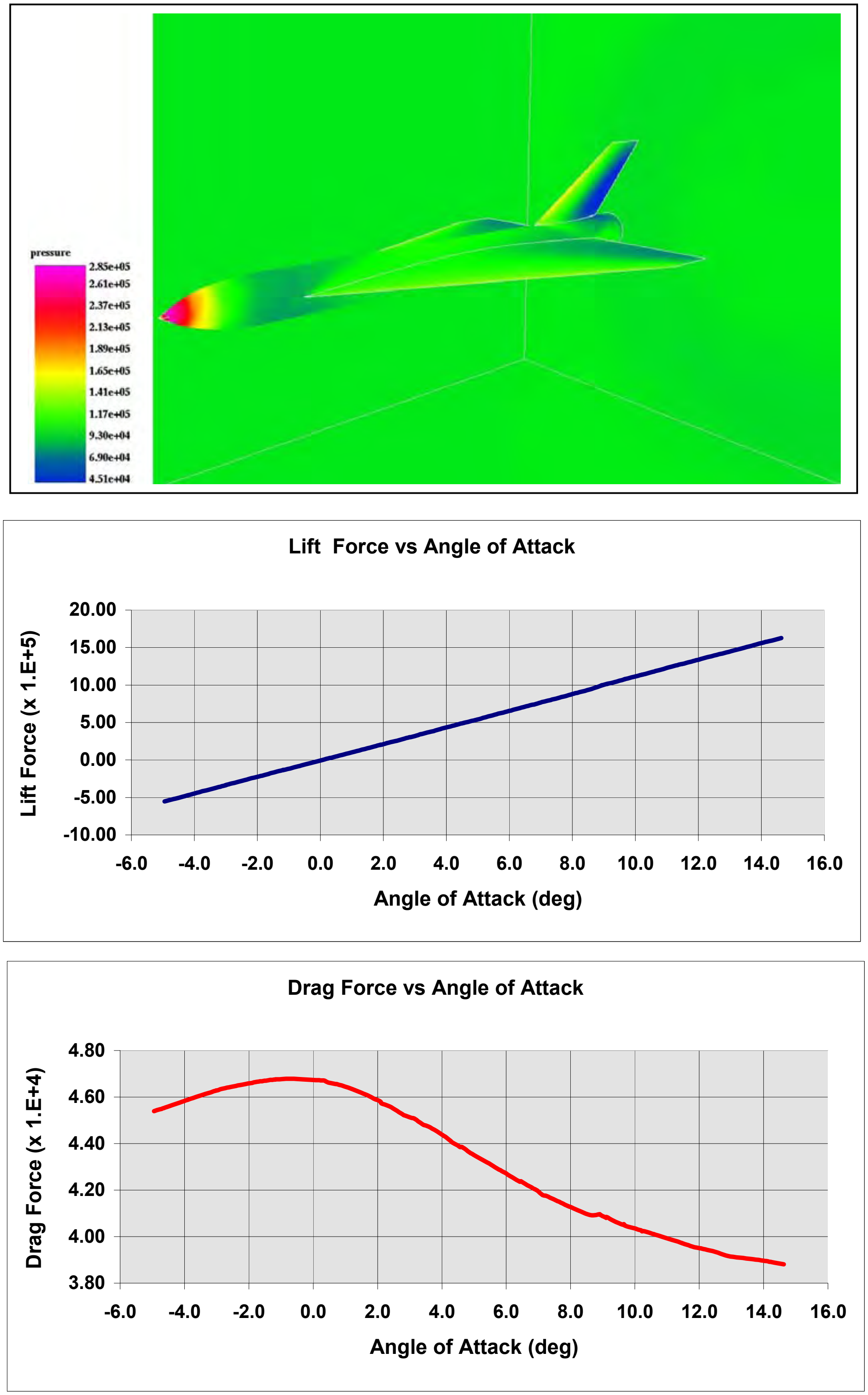
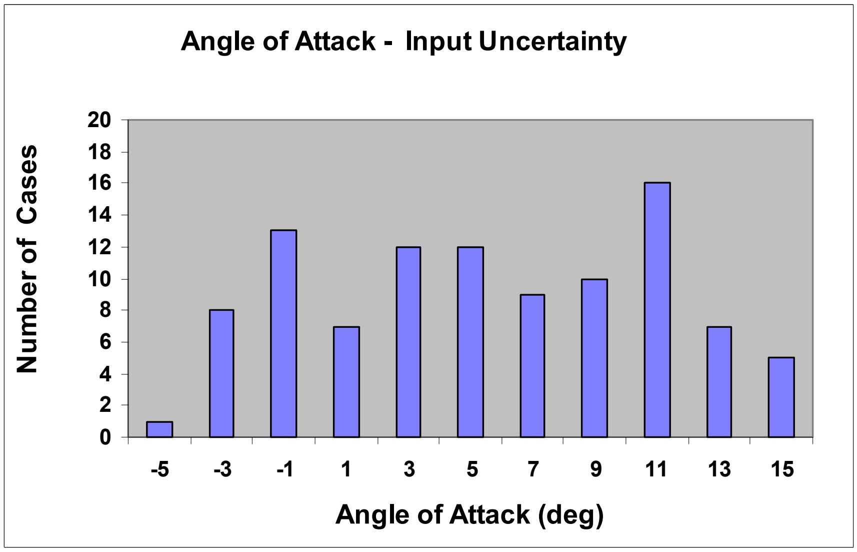
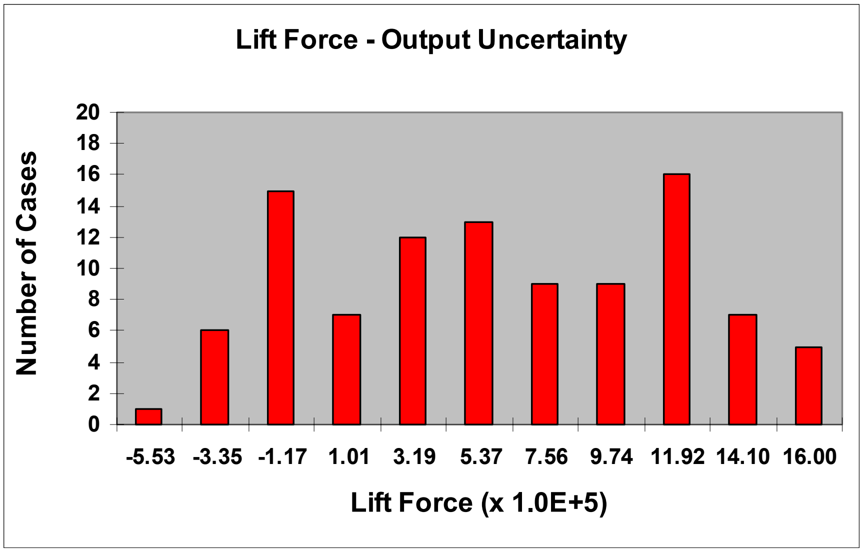


Fig. 9.4 Pressure Contours, Lift Force / Drag Force vs. Angle of Attack



angle of attack uncertainty histogram



lift force uncertainty histogram

Fig. 9.5 Histograms *Showing Uncertainty Distributions*

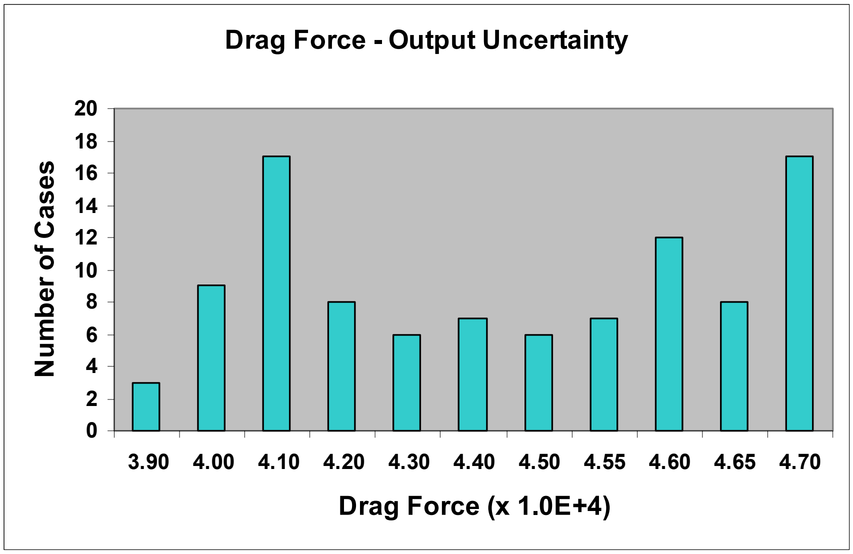
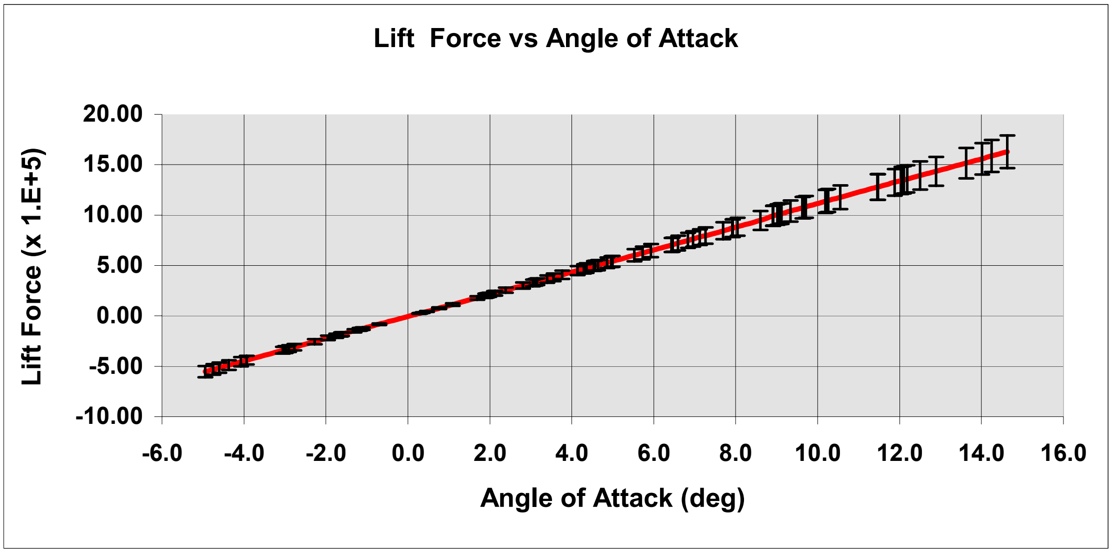


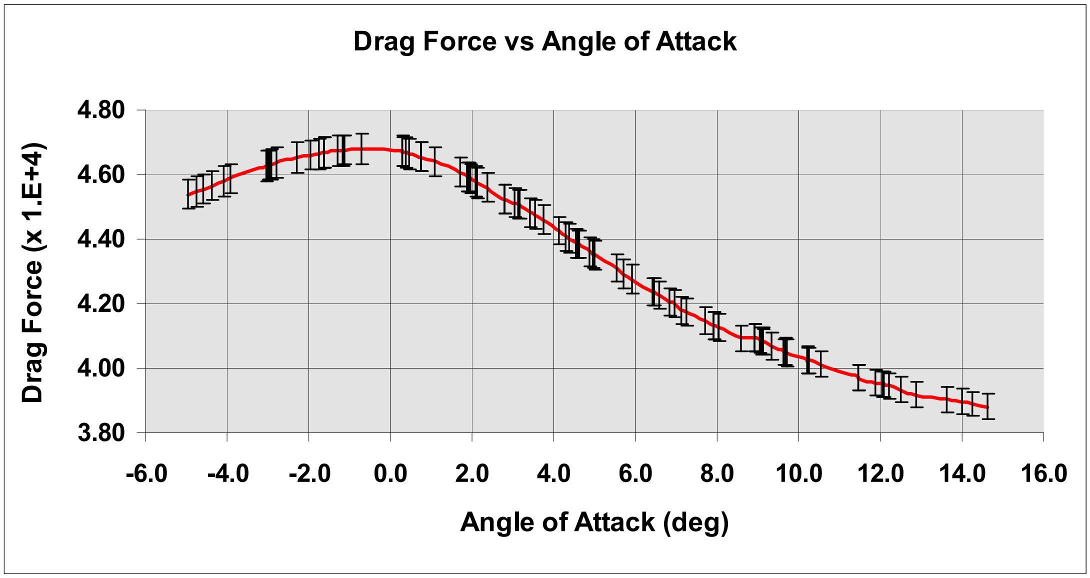
Fig. 9.6 Drag Force Uncertainty Histogram

Table 9.1 Summary Statistics

Statistic	AOA	Lift Force	Drag Force
Mean	4.46	4.91E+05	4.35E+04
Standard Error	0.53	5.87E+04	2.69E+02
Median	4.45	4.84E+05	4.40E+04
Mode	2.81	3.01E+05	4.67E+04
Standard Deviation	5.26	5.87E+05	2.69E+03
Sample Variance	27.68	3.44E+11	7.23E+06
Kurtosis	-1.03	-1.04E+00	-1.48E+00
Skewness	0.00	2.54E-02	-2.66E-01
Range	19.58	2.18E+06	7.98E+03
Minimum	-4.95	-5.53E+05	3.88E+04
Maximum	14.63	1.63E+06	4.68E+04
Sum	445.92	4.91E+07	4.35E+06
Count	100	100	100



10 % error bands on lift force



1 % error bands on drag force

Fig. 9.7 Lift and Drag Force vs. Angle of Attack with Error Bands

9.5 References

1. Oberkampf, W.L. and F.G. Blottner, "Issues in Computational Fluid Dynamics Code Verification", AIAA J., 36 (1998), pp 687-695.

2. Oberkampf, W.L., J. Helton, and K. Sentz, "Mathematical Representation of Uncertainty", AIAA Paper 2001-1645, Seattle, WA, 2001.

3. Ghanem, R.G. and P.D. Spanos, Stochastic Finite Elements: A Spectral Approach, Springer-Verlac, New York, 1991.

4. Walters, R.W. , and Luc Huyse, "Uncertainty Analysis for Fluid Mechanics with Applications", NASA / CR 2002-211449, ICASE Report No. 2002-1, NASA, February 2002.

5. Dahlquist, G, and 9. Bjorck, "Numerical Methods", Series in Computational Methods in Mechanics and Thermal Science, Prentice-Hall, 1974.

6. Hammersley, J.M., and D.C. Handscomb, "Monte Carlo Methods", Method Monographs on Applied Probability and Statistics, Fletcher and Sons LtD, Norwich, 1964.

7. Huyse, L., "Free-Form Airfoil Shape Optimization Under Uncertainty Using Maximum Expected Value And Second-Order Moment Strategies", ICASE Report 2001-18 / NASA CR 2001-211020, 2001.

8. Ghanem, R. "Stochastic Finite Elements with Multiple Random Non-Gaussian Properties", J. Eng, Mech., (1999), pp 26-40.

9. Ghanem, R., and P.D. Spanos, "Polynomial Chaos in Stochastic Finite Elements, J. Appl. Mech., (1990), pp 197-202.

10. Huyse, L. and R. W. Walters, *Random field solutions including boundary condition uncertainty for the steady-state generalized Burger equation,* ICASE Report 2001-35 / NASA CR 2001-211239, 2001.

11. Coleman, H.W. and W.G. Steel, *Experimentation and Uncertainty Analysis for Engineers*, Second Edition, Wiley & Sons, New York, 1999.

12. Robertson, S.J., L.W. Spradley, and R. Löhner, "Fluid-Structure-Thermal Interaction for Hypersonic Vehicle Analysis", Final Report, NAS1-97002, NASA Langley Research Center, 1999.

10.0 FLUID – STRUCTURE INTERACTION WITH RIGID BODY MOTION

10.1 Methodology

To compute rigid body motions, we use the arbitrary Lagrangean-Eulerian (ALE) form of equations [1]:

Total velocity $V_i = V_i^E + V_i^L$

Eulerian (fluid) velocity V_i^E

Lagrangian (mesh) velocity V_i^L

Mesh movement $u_i^{n+1} = u_i^n + \Delta t\,(V_i^{n+1} - V_i^n)$

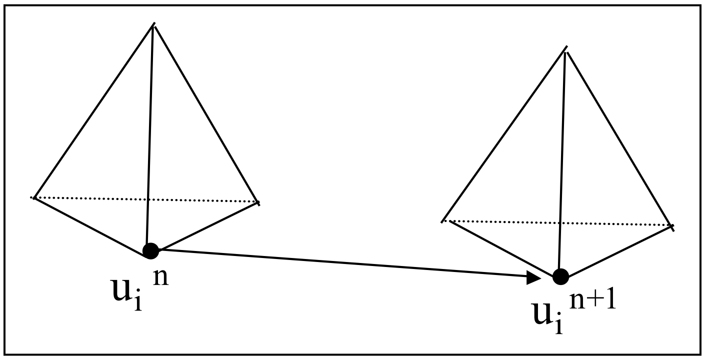


Fig. 10.1 Mesh Movement Illustrated

The following is the sequence of steps taken to loosely couple the flow and the rigid body motions [1].

- Compute body forces and moments with FEFLO
- Transform moments to the local frame of reference of the body
- Given the estimated time step, obtain the accelerations
- Given these accelerations, compute average velocities for the time-interval [t^n, t^{n+1}]
- Transform the angular velocity back from the local frame of reference of the body
- Given the time step, update the
    - positions of the points lying on the surface of the body,
    - points defining the body geometry
    - positions of the centers of mass and the rotational frame of reference

10.2 Generic HRRLS Configuration

The results ***shown*** here is for the generic HRRLS configuration #2 at Mach 2.5 and initially at 5 degrees angle of attack. The initial position is allowed to reach steady state and then the flyer is “released”. The above coupling procedure loops over time for a specified number of steps. The fluid – structure interaction loop is inside the 6 DOF loop such that the surface itself moves. The entire flyer then moves upward with the nose also tipping up as time goes by.

The solutions are shown as contours at 5 positions during the trajectory. The Mach number is shown in Fig. 10.2, the mesh velocity in Fig. 10.3 and the pressure contours in Fig. 10.4.

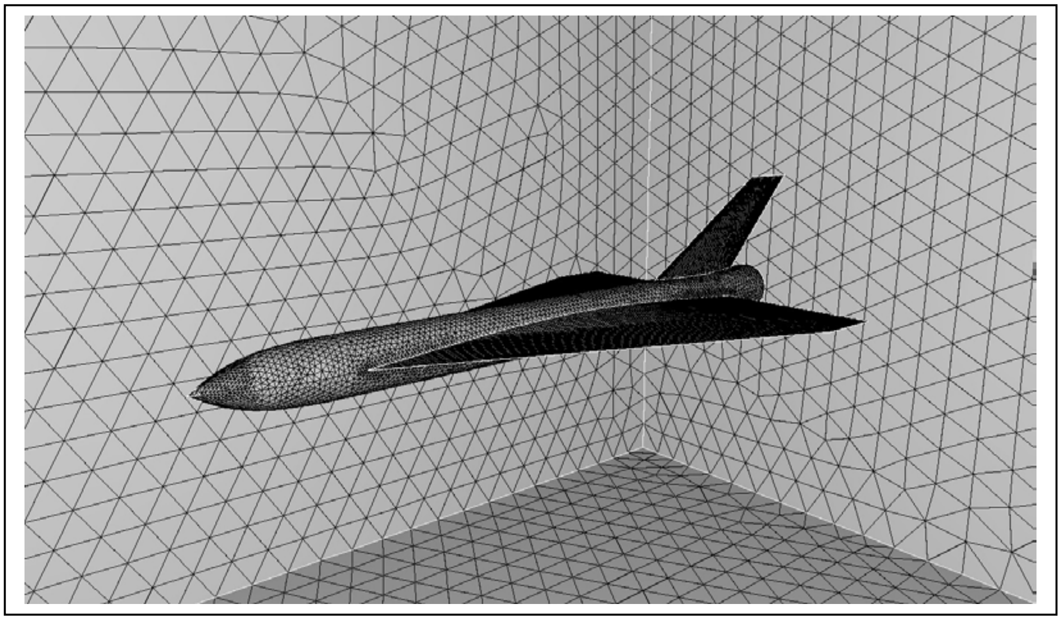


Fig. 10.1 Generic HRRLS Flyer and Surface Mesh

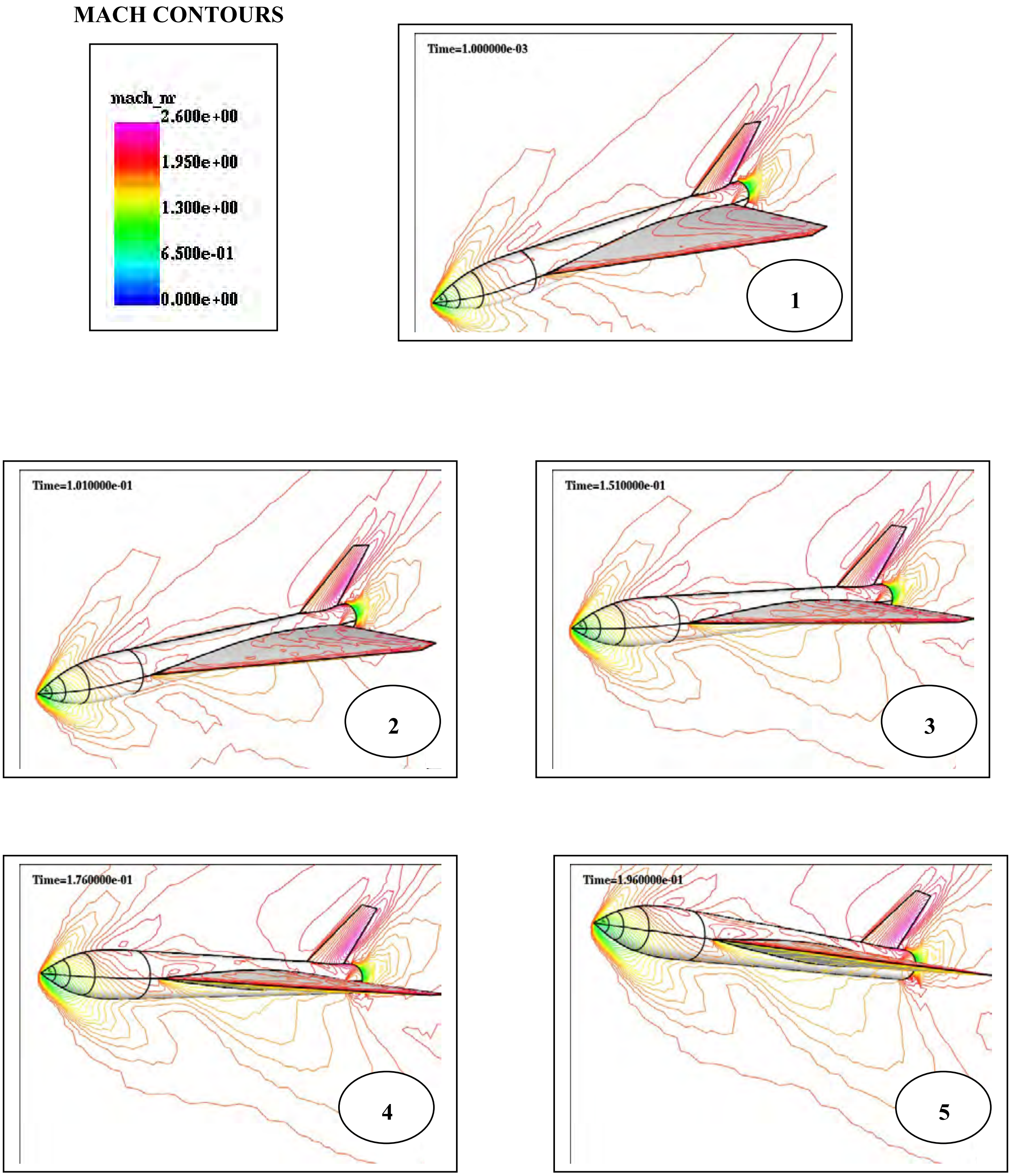


Fig. 10.2 Fluid – Structure Interaction Coupled to Rigid Body Motion

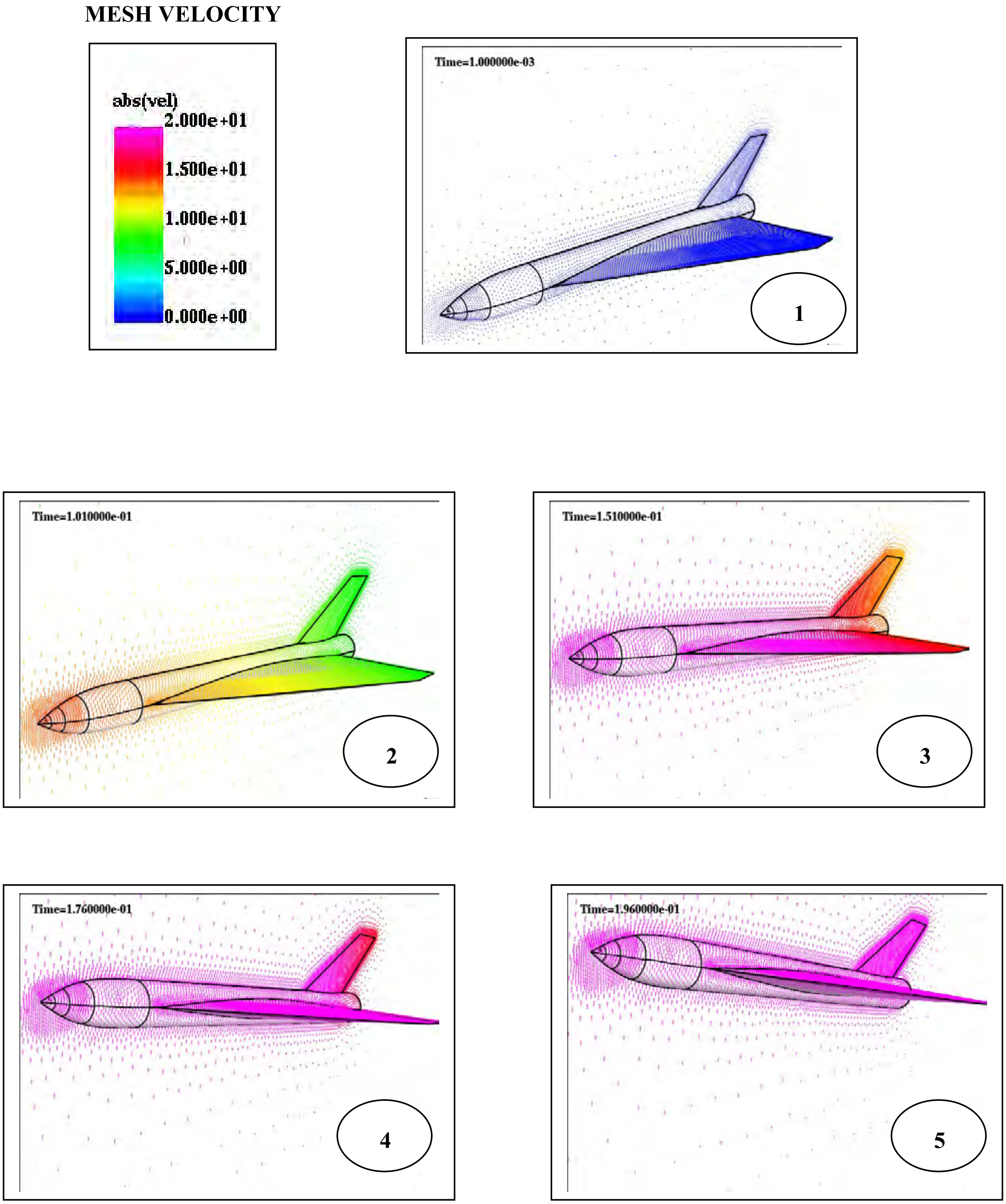


Fig. 10.3 Fluid – Structure Interaction Coupled to Rigid Body Motion

PRESSURE CONTOURS

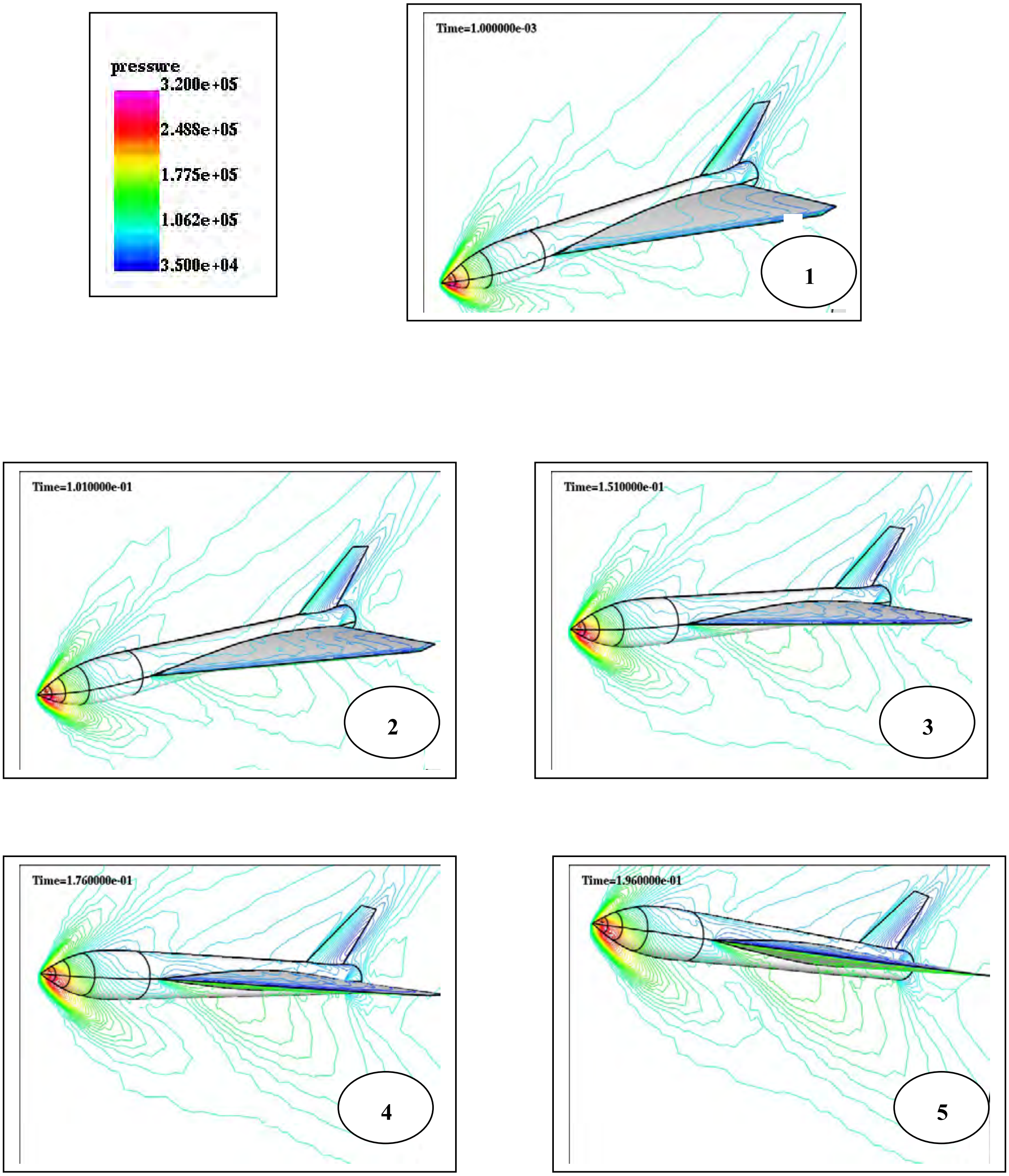


Fig. 10.4 Fluid – Structure Interaction Coupled to Rigid Body Motion

10.3 References

[1] http://www.scs.gmu.edu/~rlohner/pages/publications/fluidstructint.html